

Ștefan UNGUREANU

**Prognoza pe termen scurt
a consumului non-rezidențial
de energie electrică în contextul
conceptului smart grid**

UTPRESS
Cluj-Napoca, 2023
ISBN 978-606-737-629-6

Ștefan UNGUREANU

**PROGNOZA PE TERMEN SCURT A CONSUMULUI
NON-REZIDENȚIAL DE ENERGIE ELECTRICĂ ÎN
CONTEXTUL CONCEPTULUI SMART GRID**



UTPRESS

Cluj - Napoca, 2023

ISBN 978-606-737-629-6



Editura UTPRESS
Str. Observatorului nr. 34
400775 Cluj-Napoca
Tel.: 0264-401.999
e-mail: utpress@biblio.utcluj.ro
<http://biblioteca.utcluj.ro/editura>

Director: ing. Dan Colțea

Recenzia:

Prof.dr.ing. Călin Munteanu

Prof.dr.ing. Vasile Țopa

Prof.dr.ing. Sorin Mușuroi

Prof.dr.ing. Marcel Istrate

Conf.dr.ing. Marius Purcar

Pregătire tipar: Gabriela Groza

Copyright © 2023 Editura UTPRESS

Reproducerea integrală sau parțială a textului sau ilustrațiilor din această carte este posibilă numai cu acordul prealabil scris al editurii UTPRESS.

ISBN 978-606-737-629-6

MULȚUMIRI

În primul rând, aș dori să îi mulțumesc Domnului Profesor Vasile Țopa, pentru încrederea acordată și pentru atenția și expertiza pe care a adus-o în organizarea studiului și metodologiei de cercetare.

De asemenea, această lucrare nu ar fi fost posibilă fără sprijinul continuu și încrederea Domnului Conferențiar Andrei Cziker, încă din timpul studenției. Mulțumesc.

Aș dori să-i mulțumesc prietenului meu Radu Bindiu pentru discuțiile cu rețeaua lui neuronală și colegilor din Departamentul de Electroenergetică și Management pentru minunata lor colaborare. Aș dori în special să-i menționez pe Anca Miron, Sorin Pavel și Cosmin Darab.

Nu în ultimul rând, aș dori să îi mulțumesc soției mele Roxana pentru răbdarea și zâmbetul ei cald, iar părinților pentru sprijinul lor infinit și pentru exemplul lor de reziliență și perseverență în fața adversităților.

CUPRINS

GLOSAR DE TERMENI	3
ABREVIERI	5
PREFAȚĂ	7
INTRODUCERE	9
STADIUL ACTUAL AL CUNOAȘTERII	11
1. Rețele electrice inteligente (smart grid)	11
1.1. Definire concept	11
1.2. Controlul curbei de sarcină	16
1.3. Integrarea consumatorilor medii și mari în smart grid	23
2. Prognoza consumului de electricitate	29
2.1. Abordarea tradițională	32
2.2. Învățare automată (machine learning)	36
CONTRIBUȚII PERSONALE	41
3. Importanța temei	41
4. Ipoteze de lucru – resurse și aplicabilitate	46
5. Metodologia generală pentru prognoză	63
5.1. Prognoză bazată pe metode tradiționale	63
5.2. Prognoza bazată pe algoritmi cu învățare automată	69
5.2.1. Random Forest	72
5.2.2. Rețele neuronale artificiale	76
5.2.2.1. Perceptron multistrat (MLP)	91
5.2.2.2. Rețele neuronale recurente	93
5.2.2.3. Long-short term memory (LSTM)	96
5.2.2.4. Gated recurrent unit (GRU)	100
5.2.2.5. LSTM – encoder-decoder	105
5.2.2.6. CNN - LSTM	106
5.3. Evaluare prognoză	107

6. Prognoză agregată pentru consumatori	111
6.1 Introducere	111
6.2. Metodă și implementare	112
6.3. Rezultate	115
7. Prognoză individuală pentru consumatori	138
7.1 Introducere	138
7.2. Metodă și implementare	139
7.3. Rezultate	142
8. Evaluare prognoză pe piața de energie electrică	153
8.1. Introducere	153
8.2. Ipoteza de lucru	154
8.3. Rezultate	157
9. Discuții generale	164
10. Concluzii finale	168
10.1. Concluzii generale	168
10.2. Originalitatea și contribuțiile inovative ale tezei	174
10.2.1. Distincția față de literatura anterioară	174
10.2.2. Limitări și studii viitoare	175
REFERINȚE	177
LISTA FIGURILOR	189
LISTA TABELELOR	191
ANEXE	192
LISTĂ DE PUBLICAȚII	259

GLOSAR DE TERMENI

Agregator	Operator care adună curbele de sarcină ale mai multor consumatori sau producția de electricitate din mai multe surse în vederea vânzării sau achiziției de energie electrică.
Analiza datelor (data analytics)	Analiza datelor este știința inspectării, curățării, transformării și modelării datelor pentru a extrage informații din surse brute.
Big Data	Set sau volum de date foarte mare și complex care nu poate fi gestionat de un software tradițional de procesare a datelor.
Controlul curbei de sarcină (demand response)	Un program de control al consumului de energie care permite consumatorului să își reducă costul cu energia și în acelaș timp să contribuie la implementarea rețelelor electrice inteligente.
Curbă de sarcină	Set de valori care indică variația în timp a sarcinilor electrice, pe o perioadă determinată pentru energie electrică activă sau reactivă, înregistrate la intervale de timp consecutive și egale.
Furnizare de electricitate	Activitate comercială de vânzare și achiziție a energiei electrice către clienți finali sau alți participanți la piața de energie electrică.
Inteligență artificială	Teoria și dezvoltarea sistemelor informatice capabile să îndeplinească sarcini care necesită în mod normal inteligența umană, cum ar fi percepția vizuală, recunoașterea vorbirii, luarea deciziilor și traducerea lingvistică.
Învățarea automată (Machine learning)	Învățarea automată este o ramură a inteligenței artificiale (IA) în care sistemele informatice învață să acționeze și să se adapteze la date noi fără a fi efectiv programate.
Monitorizare inteligentă (Smart metering)	Monitorizarea datelor de consum cu ajutorul sistemelor centralizate, stocarea și transmiterea automată a acestora către furnizorul de energie și distribuitor.
Operator al pieței de energie electrică	Operatorul care asigură organizarea și administrarea pieței centralizate, cu excepția pieței de echilibrare, în vederea tranzacționării angro de energie electrică pe termen scurt, mediu și lung.
Operator de distribuție	Operatorul care deține o rețea electrică de distribuție și care răspunde de exploatarea, de întreținerea și de dezvoltarea rețelei de distribuție.
Operator de transport	Operatorul care răspunde de operarea, întreținere, dezvoltarea rețelelor de transport și interconexiunea acestora cu alte sisteme electroenergetice.

Piața de echilibrare	Piață centralizată și obligatorie pentru toți titularii de licență (de producere, transport/distribuție, furnizori de energie electrică). Pe această piață, participanții tranzacționează cantitățile de energie rezultate din abaterilor prognozei.
Piața de electricitate	Electricitatea este o marfă care poate fi cumpărată, vândută și comercializată. O piață a electricității este un sistem care permite achizițiile, prin oferte de cumpărare; vânzări, prin oferte de vânzare.
Piața pentru ziua următoare	Piață voluntară pentru furnizori și producători care facilitează tranzacționarea pe fiecare interval orar pentru ziua următoare. Participarea la această piață permite optimizarea echilibrării portofoliului de energie.
Predicție	Predicția este un act empiric care indică ce se va întâmpla în viitor cu sau fără informații prealabile.
Prognoză	Prognoza este procesul de estimare a valorilor viitoare bazate pe date trecute și analize statistice. Estimarea unei variabile de interes la o dată viitoare specificată.
Rețele electrice inteligente (smart grid)	Un concept care poate integra în mod inteligent acțiunile tuturor participanților la piața de energie electrică- producători, operatori de sistem, furnizori și consumatori - pentru creșterea eficienței energetice și asigurarea dezvoltării durabile în domeniu.
Serii de timp	Seriile de timp sunt secvențe de date care apar în ordine succesivă pe o anumită perioadă de timp.
Sistem electroenergetic național (SEN)	Sistem electroenergetic național constituie infrastructura de bază utilizată în comun de participanții la piața de energie electrică.
Prosumator	Client final care deține instalații de producere a energiei electrice, inclusiv în cogenerare, a cărui activitate specifică nu este producerea energiei electrice, care consumă și care poate stoca și vinde energie electrică din surse regenerabile.

ABREVIERI

ADAM	ADaptive Moment estimation
AMI	Advanced Metering Infrastructure (Infrastructură inteligentă de monitorizare)
AR	Auto-Regresie
ARIMA	Autoregressive Integrated Moving Average (Autoregresie diferențiată și medie mobilă)
ARMA	Autoregressive Integrated Moving Average (Autoregresie și medie mobilă)
CNN	Convolutional Neural Networks (Rețele neuronale convoluționale)
DG&S	Distributed Generation & Storage (Producție distribuită și stocare)
DL	Deep Learning (Învățare adâncă)
DR	Demand Response (Controlul curbei de sarcină)
DSM	Demand Side Management (Managementul consumului)
DT	Decission Trees (Arbore decizional)
E-Mob	Electric Mobility (Mobilitate Electrică)
ES	Exponential Smoothing (Netezire exponențială)
GD	Metoda Gradientului Descendent
GDS	Metoda Gradientului Descendent Stochastic
GRU	Gated Recurrent Unit (Unități recurente cu porți)
IA	Inteligență Artificială
IoT	Internet of things (Internetul lucrurilor)
LS-RES	Large-Scale Renewable Energy Sources (Surse mari de energie regenerabilă)
LSTM	Long-Short Term Memory (Memorie pe termen lung-scurt)
MA	Moving Average (Medie mobilă)
MAE	Mean Absolute Error (Eroare medie absolută)
MAPE	Mean Absolute Percentage Error (Eroare medie procentuală absolută)
ML	Machine Learning (Învățare automată)
MLP	Multi-Layer Perceptron (Perceptron cu straturi multiple)
MSE	Mean Squared Error (Eroarea medie pătratică)
OD	Operator de Distribuție
OTS	Operator de Transport și Sistem

PCS	Prognoza Curbei de Sarcină (Load forecasting)
PCSM	Prognoza Curbei de Sarcină pe termen Mediu
PCSS	Prognoza Curbei de Sarcină pe termen Scurt
PE	Piața de Echilibrare
PI	Piața Intra-zilnică
PZU	Piața pentru Ziua Următoare
RES	Renewable Energy Sources (Resurse regenerabile de energie)
RF	Random Forest
RL	Regresie Liniară
RML	Regresie Multi-Liniară
RMSE	Root Mean Squared Error (Rădăcina pătrată a erorii medii pătratice)
RNA	Rețea Neuronală Artificială
RNN	Recurrent Neural Networks (Rețele neuronale recurente)
SARIMA	ARIMA cu caracter Sezonier
SARIMAX	SARIMA cu factori exogeni
SEN	Sistem Electro-energetic Național
SG	Smart Grid (Rețea electrică inteligentă)
SM	Smart Metering (Contorizare inteligentă)
SNM	Smart Network Management

PREFAȚĂ

Consumul de electricitate are un impact major asupra dezvoltării sectorului economic. Accesul la energie ieftină și reglementări legislative orientate spre protecția consumatorului determină un nivel ridicat de prosperitate pentru societate. Sistemul electroenergetic asigură producția, transportul și distribuția de electricitate către utilizatori. Acest domeniu este într-o continuă schimbare, iar începând cu anul 2020 Comisia Europeană (CE) dorește reducerea semnificativă a emisiilor de CO₂ prin implementarea conceptului de neutralitate climatică [1]. Producția de electricitate este a doua cea mai mare sursă de CO₂, după sectorul transporturilor. Menținerea necesarului actual de electricitate cu surse neconvenționale sau regenerabile este o provocare care momentan nu are o soluție fiabilă și durabilă.

Îmbunătățirea și automatizarea prognozelor de consum ajută la această tranziție tehnologică. Anticiparea cu succes a variațiilor de consum contribuie la echilibrarea balanței producție-consum și la menținerea stabilității rețelei electrice, atât din punct de vedere tehnic cât și financiar. Activitatea de realizare a prognozelor de consum necesită cunoștințe aprofundate și personal instruit adecvat. În cadrul companiilor private astfel de sarcini încarcă activitatea angajaților, generând costuri adiționale și având un impact negativ asupra rezultatelor prognozei. Din acest motiv, această lucrare abordează prognoza de electricitate cu algoritmi bazați pe învățare automată, care facilitează integrarea consumatorilor non-rezidențiali în rețele electrice inteligente (Smart Grid).

În primul și al doilea capitol al lucrării se prezintă stadiul actual al cunoașterii din punct de vedere al rețelelor electrice inteligente și al prognozei de energie electrică. În urma studiului literaturii de specialitate s-au identificat și centralizat cele mai importante lucrări pe tema prognozei de consum realizate atât cu metode tradiționale cât și cu metode specifice inteligenței artificiale. Scopul urmărit este încadrarea metodologiei și algoritmilor implementați în prezenta lucrare în raport cu studiile recunoscute de comunitatea științifică.

În capitolul 3 se evidențiază importanța lucrării în contextul actual de evoluție al pieței de energie electrică din România pentru prezentarea contribuției personale. În capitolul 4 se prezintă resursele și datele folosite pentru implementarea algoritmilor de prognoză pe termen scurt. Capitolul 5 prezintă fiecare algoritm implementat și

metodologia generală urmată pentru realizarea prognozelor propuse. Au fost utilizate mai multe metode tradiționale de prognoză pentru a oferi o bază solidă de comparație care să justifice folosirea algoritmilor bazați pe învățare automată.

Capitolul 6 și 7, prezintă implementarea algoritmilor pe baza datelor de consum obținute de la un grup de consumatori industriali și comerciali. Sunt implementate două abordări, una în care se prognozează consumul individual pentru fiecare consumator (ulterior fiind însumate prognozele) și una în care se prognozează consumul agregat. În acest mod s-a urmărit identificarea algoritmului care asigură cea mai mică eroare, și cu un impact minim pe piața de energie electrică. În practică, furnizorii de energie electrică realizează prognoze individuale pentru marii consumatori din portofoliu și prognoze agregate pentru consumatori mici. O altă practică des întâlnită este delegarea responsabilității de echilibrare către un alt participant din piața de echilibrare care are un portofoliu compus din mulți consumatori. Astfel se obține o curbă de sarcină agregată cu variații și fluctuații în timp mai mici, față de seriile de timp generate de un consumator individual.

Pentru evaluarea prognozelor efectuate în capitolul 8 se abordează o metodă nouă care calculează în mod real impactul prognozelor de consum pe piața de energie. Dincolo de indicatorii statistici care evidențiază măsura erorilor, se cuantifică în termeni financiari impactul generat de fiecare algoritm de prognoză conform scenariilor prezentate.

Capitolul 9 prezintă concluziile generale și discută rezultatele prognozelor efectuate. S-a urmărit îndeaproape ca toată metodologia lucrării să fie în acord cu practica specifică industriei energetice și necesitățile implementării conceptului de rețele electrice inteligente. În Capitolul 10 se evidențiază originalitatea și contribuțiile inovatoare aduse prin prezenta teză de doctorat.

INTRODUCERE

Consumatorii industriali și comerciali de electricitate pot deveni participanți activi în sistemul electroenergetic cu ajutorul tehnologiilor promovate prin conceptul rețelelor electrice inteligente (SG). Prognoza variațiilor de consum pe termen scurt (zilnic, orar sau minute) are potențialul de a oferi rețelelor electrice inteligente informații care pot optimiza: i) costul electricității, echilibrarea balanței producție-consum și transferul optim de electricitate între rețelele regionale; ii) reducerea consumului propriu tehnologic în rețele, identificarea rapidă a defectelor și mentenanță preventivă. Pentru a răspunde la această provocare s-a abordat cercetarea propusă prin implementarea mai multor algoritmi și modele cu învățare automată pentru prognoza curbelor de sarcină. Prognoza de consum reprezintă informație care poate fi utilizată în piața de electricitate atât pentru optimizarea costurilor cât și pentru îmbunătățirea funcționării rețelelor electrice [2].

Studiul realizat în prezenta lucrare aplică metodele propuse pe datele de consum obținute de la cinci consumatori industriali din domenii diferite (prelucrarea lemnului, comerț și retail). Algoritmii de prognoză au fost implementați pe curbele de sarcină individuale cât și pe curba agregată a întregului grup de consumatori. Aceste abordări imită modul în care un furnizor își gestionează portofoliul de clienți cu respectarea obligațiilor de echilibrare a electricității achiziționate și vândute. Agregarea curbelor de sarcină poate ajuta la obținerea unor erori de prognoză mai mici și facilitează strategiile de control ale curbei de sarcină datorită rezervei de putere disponibilă. Rezultatele obținute au fost analizate prin compararea prognozelor individuale însumate cu cea agregată, iar în final s-a identificat metoda care generează cea mai mică eroare de prognoză. Astfel argumentele justificative pentru atragerea mediului privat spre zona rețelelor electrice inteligente sunt prognoza consumului și controlul curbei de sarcină. O rețea electrică inteligentă reprezintă o rețea electrică optimizată în care se atinge un nivel ridicat de eficiență energetică printr-o comunicare rapidă, în timp real, între toți participanții la piața de energie. Prima etapă a cercetării s-a adresat dezvoltării algoritmilor cu învățare automată (machine learning- ML) pentru prognoza seriilor de timp (curbe de sarcină). Mediul de programare utilizat pentru dezvoltarea algoritmilor de prognoză are o arhitectură deschisă și permite dezvoltarea de noi aplicații care să fie ușor utilizate. Implementarea algoritmilor în această

lucrare facilitează o gamă largă de aplicații pentru prognoza consumului de energie electrică, deschizând direcția spre un grad ridicat de automatizare necesar conceptului de SG.

În prezent colaborarea dintre participanții la piața de energie electrică cu consumatorii de electricitate se rezumă la un contract comercial de furnizare. Colaborarea se materializează prin negociere, semnarea contractului și achitarea facturii care de cele mai multe ori are la bază un tarif monom (lei/MWh). Acest preț este dimensionat pentru a compensa lipsa de comunicare și planificare a consumului de electricitate. Costul electricității este perceput ca fiind incontrolabil de către factorul decizional din mediul privat, ceea ce creează o multitudine de probleme în rețelele electrice (atât tehnice cât și financiare). În această lucrare se dorește construirea de modele pentru prognoză care să *învețe automat din variațiile curbelor de sarcină în timp și obținerea de prognoze relevante pentru mediul concurențial*. Având în vedere noile reglementări legislative din 2021 prin care se modifică decontarea electricității la nivelul de 15 minute¹, munca necesară pentru prelucrare, analiză și prognoză crește de patru ori, motiv pentru care automatizarea procesului de prognoză a consumului de energie devine una deosebit de importantă.

Fundamentul lucrării este bazat pe implementarea algoritmilor cu învățare automată pentru prognoza curbelor de sarcină electrică și evaluarea financiară a erorilor pe piața de energie. Algoritmii sunt aplicați pe baze de date obținute de la parteneri industriali și comerciali. Algoritmii ML implementați pentru prognoza consumului de electricitate a consumatorilor medii și mari obțin erori medii procentuale absolute (MAPE) în jurul valorii de 5%. Validarea algoritmilor se face prin indicatori specifici și calculul riscului (impactului) pe piața de energie. Prognoza pe termen scurt a sarcinii electrice predetermină funcționarea sistemelor energetice, deoarece producția de energie electrică trebuie să susțină cererea în orice moment și cu orice cost. Decontarea costurilor dezechilibrelor de energie electrică pe piața de echilibrare este obligatorie.

¹<https://www.anre.ro/ro/presa/comunicate/comunicat-privind-testarea-aplicatiilor-informatic-destinate-aplicarii-intervalului-de-decontare-de-15-minute>

STADIUL ACTUAL AL CUNOAȘTERII

1. Rețele electrice inteligente (smart grid)

1.1. Definiere concept

Creșterea eficienței energetice, accelerarea producției de energie regenerabilă și dezvoltarea tehnologiilor care ajută la gestionarea rețelelor electrice sunt în topul priorităților pe care le au companiile energetice din întreaga lume, potrivit studiilor prezentate în [2] și [3]. România are un drum lung de parcurs pentru a se alinia la tendințele europene de digitalizare în sectoarele energetice. Energia electrică poate reprezenta un business lucrativ pentru consumatori, iar tehnologiile smart grid (SG) au capacitatea să transforme consumatorii în resurse energetice care să contribuie la minimizarea costurilor și echilibrarea rețelelor de distribuție.

Conceptul de smart grid se aplică pentru modernizarea rețelelor actuale de electricitate, astfel încât să facă față efortului global de reducere al emisiilor de CO₂ în contextul încălzirii globale. SG poate fi definit printr-un „IoT” al energiei sau „*Internet al energiei*”, deoarece pune la dispoziția participanților de pe piața de energie informații în timp real, oferind astfel posibilitatea unor alegeri inteligente, benefice pentru întregul sistem energetic. Tehnologia aferentă SG constă într-un ansamblu de sisteme de control și management al rețelei, de senzori și mijloace de comunicare și informare, care încorporează atât elemente tradiționale, cât și de ultimă generație. SG combină elemente de software și hardware menite să îmbunătățească semnificativ modul în care este operat sistemul energetic actual [4], oferind în același timp și posibilitatea modernizării ulterioare. Prin urmare, SG nu presupune înlocuirea rețelei electrice existente, ci urmărește îmbunătățirea funcționării actuale prin creșterea nivelului de transparență în sistem (monitorizare și control). Rețelele electrice inteligente pot optimiza integrarea producției din surse regenerabile. În plus, consumatorii vor avea posibilitatea de reducere a vârfului de sarcină, adaptând puterea absorbită din rețea la nevoile fluxului tehnologic de producție, ținând cont de condițiile și resursele energetice disponibile. Tehnologiile SG au capacitatea să transforme consumatorul într-o resursă de energie care

poate să sprijine rețeaua electrică prin modificarea curbei de sarcină (Figura 1.1).

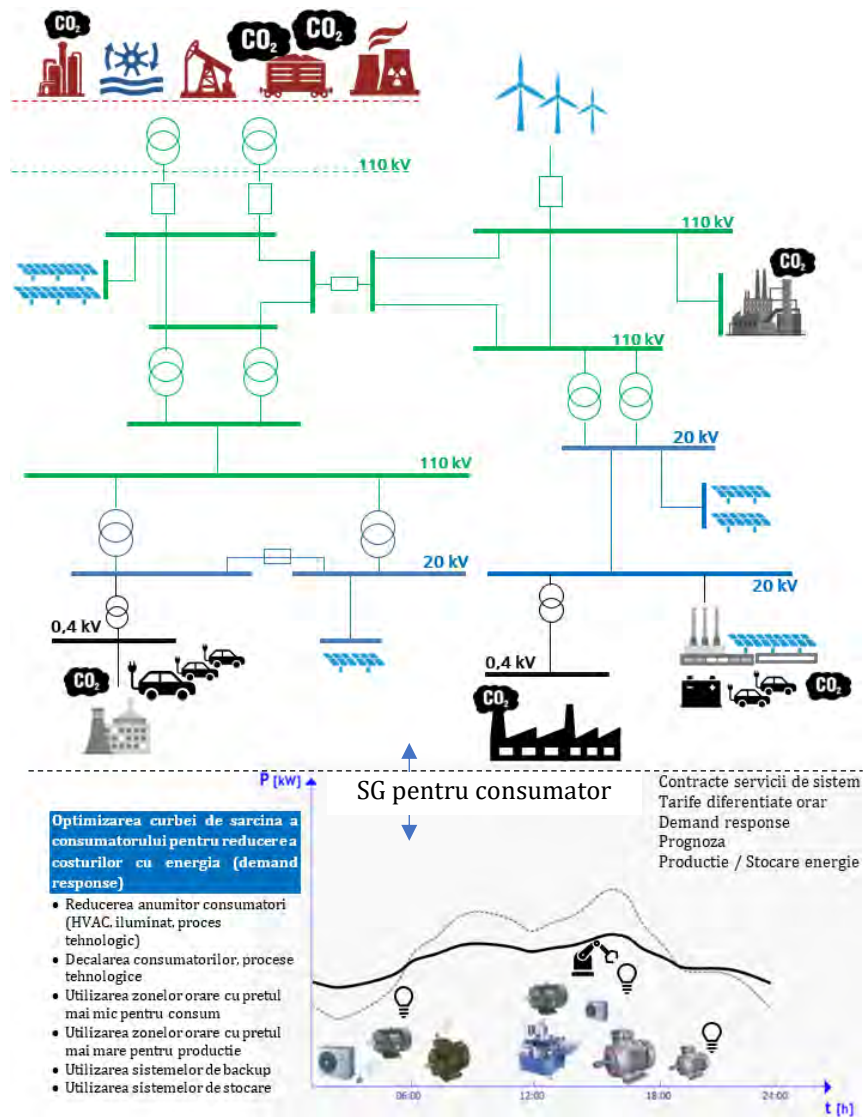


Figura 1.1 Controlul consumatorului industrial în rețele electrice inteligente

Orice variație sau schimbare în comportamentul oricărui participant din Sistemul Electroenergetic Național (SEN) generează o reacție care afectează întregul sistem energetic. În acest context, operatorului de transport și sistem din România (Transelectrica) îi revine sarcina extrem de complexă și costisitoare de a asigura echilibrarea rețelei electrice naționale în orice moment.

Cele mai vizate domenii de dezvoltare în domeniul energetic din punct de vedere al investițiilor în Europa sunt cele legate de managementul inteligent al rețelelor (Smart network management - SNM) cu 26%, managementul consumului (Demand Side Management - DSM) cu 24% și respectiv integrarea producției distribuite și stocare (Distributed Generation and Storage - DG&S) cu 26%. Toate reprezentând împreună aproximativ 76% din investițiile totale în rețehnologizarea rețelelor electrice, conform datelor prezentate în [5] până în anul 2015. În Figura 1.2 se prezintă informații suplimentare despre finanțarea pe fiecare domeniu specific SG. În toate domeniile cea mai mare investiție provine din resurse private 54%, în timp ce restul este împărțit între finanțări naționale de 22% și finanțare din partea UE, 24%. O analiză suplimentară indică faptul că o parte importantă din investițiile private provin de la companii comerciale - cum ar fi companiile IT și producătorii de tehnologie - indicând un interes semnificativ al sectorului privat pentru soluțiile SG. Un procent de 20% din totalul investițiilor private este finanțat prin stimulente specifice legislației europene disponibile operatorilor de rețea pentru activități de inovare.

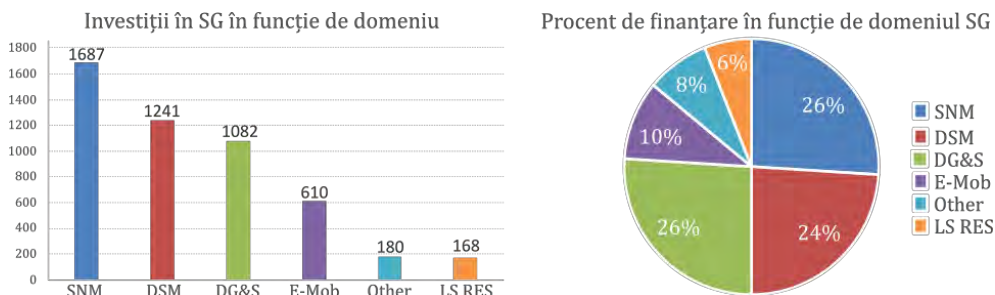


Figura 1.2 Stadiul de finanțare al SG în UE la nivelul anului 2015²

Operatorii de Distribuție sunt entitățile comerciale care investesc cele mai multe resurse, urmate de universități și producători de tehnologie. Conform rapoartelor realizate de Centrul Comun de Cercetare (JCR) al Comisiei Europene și Directoratul General pentru Energie care poate fi consultat în [5] s-au identificat cinci domenii principale pentru SG:

² <https://ses.jrc.ec.europa.eu/smart-grids-observatory>

- a) **SNM:** Proiectele din acest domeniu se concentrează pe creșterea flexibilității operaționale a rețelei de energie electrică prin capacități îmbunătățite de monitorizare și control al rețelei. De obicei, aceasta implică instalarea de echipamente de monitorizare și control a rețelei și comunicații de date rapide și în timp real. Exemple de aplicații:
- sisteme de monitorizare complexe la nivelul rețelei de transport.
 - dispozitive de măsurare detaliată și interfețe avansate pentru prosumatori la nivelul rețelei de distribuție.
 - instrumente pentru observabilitatea rețelei europene.
 - instrumente pentru evaluarea fiabilității rețelei paneuropene.
 - senzori avansați pentru rețele care identifică anomalii și le comunică.
 - instrumente pentru controlul și remedierea defectelor în rețele electrice.
 - noi capacități pentru controlul frecvenței, controlul reactiv și controlul fluxului de putere.
 - stații de distribuție controlabile, invertoare inteligente și selectivitate inteligentă a protecțiilor.
- b) **DSM:** Acest domeniu include atât proiecte care au ca scop mutarea consumului într-un alt moment (controlul curbei de sarcină), cât și proiecte care vizează reducerea nivelului de consum fără a influența procesul și fără a afecta nivelul de confort (conservarea energiei / eficiența energiei). Exemple de aplicații:
- dezvoltarea de soluții și servicii pentru controlul curbei de sarcină și eficiență energetică.
 - implementarea inițiativelor și soluțiilor pentru a încuraja consumatorii rezidențiali, comerciali și industriali de a modifica nivelul și tiparul de utilizare a energiei electrice.
 - abilitarea consumatorilor de energie electrică (inclusiv a consumatorilor vulnerabili) prin punerea în aplicare a serviciilor de măsurare inteligentă și a inițiativelor de sensibilizare.
 - controlul curbei de sarcină și gestionarea energiei.
- c) **DG&S:** Acest domeniu include proiecte axate pe scheme de control avansat și noi soluții pentru integrarea producției distribuite (DG) și stocării energiei în rețeaua de distribuție, asigurând în același timp fiabilitatea și securitatea sistemului. Exemple de aplicații:

- instrumente de planificare și analiză a rețelei electrice pentru evaluarea capacității sale pentru conexiunile DG.
 - suport activ al rețelei electrice (controlul frecvenței și tensiunii) prin invertoare inteligente care să faciliteze conexiunea la DG.
 - arhitecturi de control centralizate vs. descentralizate.
 - integrarea sistemelor de stocare cu energie electrică pentru utilizare ulterioară.
 - integrarea stocării distribuite a energiei electrice pentru a crește flexibilitatea operațională a rețelei de distribuție.
 - dezvoltarea conceptului de date accesibile și interoperabile și a soluțiilor de automatizare pentru integrarea DG&S.
 - agregarea DG controlabilă și stocarea în centrale electrice virtuale și micro-rețele.
- d) **E-mobilitate:** Proiectele din acest domeniu se axează pe integrarea inteligentă a vehiculelor electrice (EV) și a vehiculelor hibride plug-in (PHEV) în rețeaua electrică. Exemple de aplicații:
- dezvoltarea infrastructurii de încărcare inteligentă și a strategiilor de control.
 - integrarea EV pentru furnizarea de servicii auxiliare.
 - dezvoltarea și validarea interfețelor vehicul-to-grid (V2G).
- e) **LS_RES:** Proiectele din acest domeniu vizează în principal integrarea resurselor regenerabile de energie (Renewable Energy Sources) - RES la rețeaua de transport sau de distribuție de înaltă tensiune. Exemple de aplicații:
- dezvoltarea și testarea noilor tehnologii de rețea care să permită creșterea capacității și flexibilității rețelei la nivel paneuropean, menținând în același timp fiabilitatea sistemului.
 - rețele electrice off-shore pentru integrarea energiei eoliene.
 - dezvoltarea unei platforme de testare numerică pentru testarea și validarea noilor modele de piață pentru integrarea producției flexibile masive dispersate pe mai multe piețe regionale de energie.
 - dezvoltarea de noi tehnologii împreună cu un sistem inovator pentru abordarea managementului pentru furnizarea de servicii

de sistem (controlul tensiunii și frecvenței) de către parcurile eoliene agregate.

- instrumente de prognoză pentru producția de resurse regenerabile de energie (RES).
- integrarea DSM pentru furnizarea de servicii auxiliare de către DSO-uri pentru sprijini funcționarea sistemului de transport.

f) **Altele:** Restul aplicațiilor proiectului rețelei inteligente care nu sunt incluse în domeniile menționate mai sus sunt incluse în acest grup.

- piața și reglementări (de exemplu, identificarea lacunelor de cercetare și tehnologie pentru rolurile emergente și viitoare ale operatorilor de sistem în sistemul integrat european).
- securitatea cibernetică (dezvoltarea de noi mijloace de securitate pentru infrastructuri critice).

În rapoartele Centrului Comun de Cercetare (JCR) nu se face referire directă către prognoza consumului de electricitate, se menționează doar despre prognoza RES. Prin prezenta lucrare se dorește scoaterea în evidență a *importanței prognozei consumului de electricitate ca un instrument de integrare a utilizatorilor în SG.*

1.2. Controlul curbei de sarcină

Transferul de electricitate se realizează între instalațiile consumatorilor și rețeaua electrică prin intermediul echipamentelor electrice programate corespunzător, cu scopul de a satisface cererea fluctuantă de electricitate la un standard de calitate ridicat, securitate în alimentare și la un preț accesibil [6].

Echilibrul dintre producția și consumul de electricitate este dificil de menținut în contextul următoarelor tendințe în sistemele electro-energetice europene:

- integrarea surselor regenerabile implementate la scară largă.
- descentralizarea surselor de producție.
- trecerea la mobilitate electrică.
- creșterea consumului de electricitate.
- defaectarea centralelor electrice pe bază de combustibili fosili.
- condiții meteorologice extreme.

Serviciile auxiliare tradiționale furnizate de operatorul rețelei electrice facilitează și susțin fluxul continuu de energie electrică prin care

producția va satisface în mod continuu cererea. Dezvoltarea rețelelor inteligente determină schimbarea modului în care se utilizează electricitatea în așa fel încât să ajute la stabilizarea sistemului [7]. Managementul sarcinii (DSM) și controlul curbei de sarcină (Demand Response - DR) au un potențial mare de creștere a fiabilității rețelei electrice și de îmbunătățire a costurilor sistemului de operare, oferind în același timp stimulente financiare pentru consolidarea eficienței energetice pentru consumatori. Principalele provocări sunt diferitele tipuri de caracteristici ale consumatorilor care sunt dificil de modelat într-un mediu dinamic cum este rețeaua electrică. Prezenta lucrare se concentrează pe consumatorii industriali și comerciali care conțin o varietate de echipamente cu cerințe diverse de funcționare (Figura 1.3). De asemenea, anumite procese tehnologice și tipuri de echipamente similare utilizează electricitatea diferit în funcție de comportamentul angajaților, managementul companiilor sau cerințele de operare. Pentru controlul curbei de sarcină, învățarea automată poate să construiască scenarii în timp real prin care să se implementeze managementul sarcinii pentru grupuri de consumatori industriali și comerciali, având ca factor de decizie evoluția prognozei.

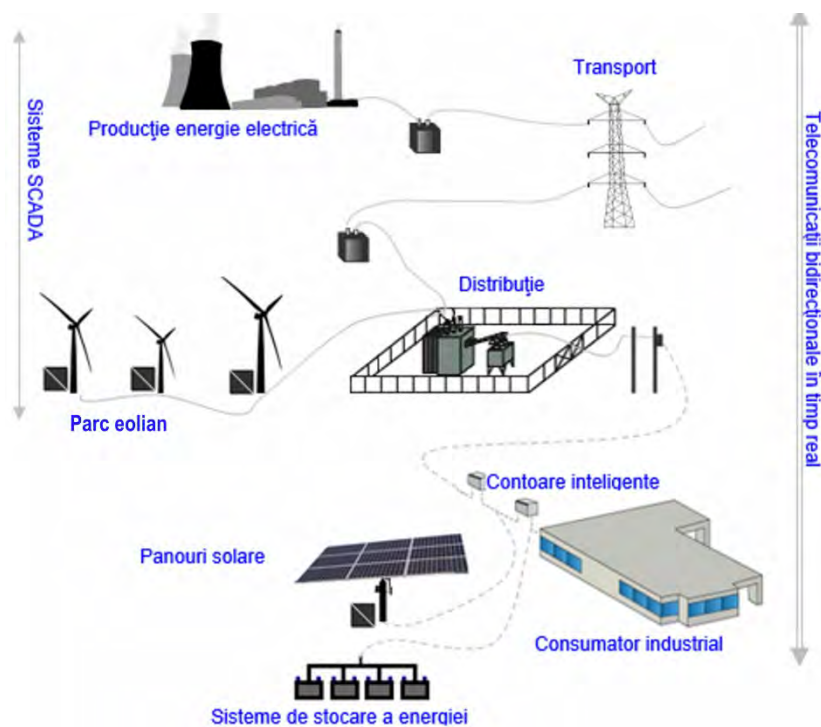


Figura 1.3 Integrarea consumatorului industrial în rețele inteligente

Una dintre principalele utilizări ale energiei electrice în sectorul industrial include acționarea sistemelor electrice, respectiv alimentarea motoarelor care acționează pompe, ventilatoare, benzi transportoare etc. Alte utilizări uzuale includ tehnologia informației (calculatoare, comunicații și echipamente de birou), iluminat interior/exterior, încălzire, încălzirea apei menajere, ventilație, aer condiționat, servicii alimentare, tratamentul apei etc. Fiecare dintre acestea oferă oportunități pentru eficientizarea consumului de energie electrice prin control și automatizare. După cum este prezentat în Tabel 1.1 există multe provocări în implementarea soluțiilor de DR.

Tabel 1.1 Impactul SG pentru piața de energie electrică

	Avantaje	Dezavantaje
Consumator	<ul style="list-style-type: none"> ▪ Automatizarea proceselor de management și audit energetic. ▪ Acces la date în timp real. ▪ Mentenanță preventivă. ▪ Eliminarea înregistrărilor eronate a datelor. ▪ Creșterea siguranței în alimentarea cu energie electrică. ▪ Îmbunătățirea calității energiei. ▪ Utilizarea și valorificarea resurselor proprii de producție a energiei. 	<ul style="list-style-type: none"> ▪ Costuri ridicate pentru implementarea infrastructurii de comunicații. ▪ Întreținerea sistemului de monitorizare și gestionarea datelor. ▪ Dificultăți ridicate de confidențialitatea datelor. ▪ Complexitatea sistemelor de monitorizare și control pot influența negativ percepția factorilor decizionali. ▪ Lipsa reglementărilor în domeniul standardelor.
Furnizor	<ul style="list-style-type: none"> ▪ Gestiunea costurilor mult mai eficient. ▪ Notificări automate (comunicare directă între sisteme). ▪ Flexibilitate ridicată pentru achiziția energiei. ▪ Protejarea mediului. ▪ Reducerea emisiilor de CO₂. 	<ul style="list-style-type: none"> ▪ Volum foarte mare de date care trebuie gestionat. ▪ Costuri ridicate cu personal calificat. ▪ Lipsa reglementărilor în domeniul standardelor.
Operator de rețea electrică	<ul style="list-style-type: none"> ▪ Diagnosticarea rapidă a problemelor din rețea. ▪ Interoperabilitatea sistemelor de control. ▪ Aplatizarea curbei de sarcină. ▪ Protejarea mediului. ▪ Reducerea emisiilor de CO₂. 	<ul style="list-style-type: none"> ▪ Costuri ridicate pentru implementarea infrastructurii de comunicații. ▪ Volum foarte mare de date. ▪ Lipsa reglementărilor în domeniul standardelor. ▪ Costuri ridicate cu personal calificat.
Producători de energie electrică	<ul style="list-style-type: none"> ▪ Facilitează toate opțiunile de generare și stocare. ▪ Resurse diverse cu conexiuni „plug-and-play”. ▪ Multiplică opțiunile pentru generarea și stocarea electrică. ▪ Noi oportunități pentru o producție mai eficientă și mai curată. 	<ul style="list-style-type: none"> ▪ Costuri ridicate pentru implementarea infrastructurii de comunicații. ▪ Volum foarte mare de date. ▪ Lipsa reglementărilor în domeniul standardelor. ▪ Costuri ridicate cu personal calificat.

În domeniul industrial și în special pentru marii consumatori de energie electrică, prețul la energie și politicile de mediu sunt principalii factori de eficiență energetică. Într-o industrie globalizată, scăderea costurilor de producție prin reducerea consumurilor energetice este una dintre acțiunile majore care asigură competitivitate. Producția și consumul de energie electrică sunt surse majore responsabile de emisii

CO₂. Ca și un actor important privind emisiile de CO₂ (36% din emisiile de dioxid de carbon sunt datorate industriilor de fabricație³), sectorul industrial este supus permanent presiunilor legislative care conduc la reducerea propriilor emisii, și a consumului de energie, în mod implicit. Această constrângere pentru protejarea mediului poate fi aplicată în diferite moduri, cum ar fi taxele de carbon (prin adăugarea unui cost pentru fiecare tonă de CO₂ emisă), sistemele de comercializare a emisiilor de CO₂ (cotă de CO₂ + cotă certificate CO₂ de schimb în piețe specifice), sau a obligațiilor de CO₂ (obligațiile de a reduce emisiile de CO₂ la un nivel fix).

Managementul consumatorului sau gestionarea sarcinii reprezintă programe și activități menite să încurajeze consumatorii să își schimbe comportamentul de consum, inclusiv perioada și nivelul consumului. În mod tradițional această schimbare înseamnă aplatizarea curbei de sarcină. Cu dezvoltarea acestui segment de rețele inteligente, consumatorul va putea să își programeze producția în așa fel încât să utilizeze energia când prețul este mic pe piață și să reducă consumul atunci când prețul este mare sau să ofere servicii de sistem în schimbul unor avantaje financiare.

În cadrul unui program de control al consumatorilor, furnizorul sau distribuitorul va notifica clienții înscriși cu "*mesaje*" care conțin posibile acțiuni pentru reducerea consumului sau suplimentarea producției de energie [8]. Aceste acțiuni sunt stabilite de comun acord între cele două părți și necesită un timp de reacție rapid (zece minute sau mai puțin), sau ar putea fi programate cu ore sau chiar zile în avans. Prin obținerea angajamentelor din partea clienților pentru a reduce sarcina, furnizorul poate planifica și dezvolta soluții eficiente pentru acoperirea vârfului de sarcină, în schimb consumatorul putând beneficia de avantaje financiare pentru punerea la dispoziție a unei capacități electrice controlabile. În Tabel 1.2 se prezintă cerințele tehnice necesare demand response (DR). Doi pași foarte importanți pentru DR sunt auditul energetic și crearea scenariilor de gestionare a sarcinii. Auditul energetic este practic cel mai important pas, deoarece stabilește disponibilitatea consumatorului de a participa în astfel de programe dar și măsurile de control care pot fi implementate.

³International Energy Agency
(<http://www.iea.org/Textbase/npsum/tracking2007SUM.pdf>)

Tabel 1.2 Cerințe tehnice

Cerințe	
Hardware	<ul style="list-style-type: none"> • contor electric inteligent. • sistem de control pentru punerea în aplicare a gestiunii energiei și a funcțiilor de automatizare. • senzori, aparatură de protecție, dispozitive inteligente (relee cu comunire conto).
Telecomunicații	<ul style="list-style-type: none"> • infrastructură de telecomunicații. • standarde și protocoale.
Monitorizare	<ul style="list-style-type: none"> • obținerea de informații despre energie ieftină și din surse regenerabile. • stocarea energiei generate prin mijloace proprii și gestionarea consumului în timp real. • funcționalitate “Plug-and-play” pentru integrarea de sarcini noi în sistemul de management. • detectarea defectelor. • identificarea deteriorării aparatelor. • diagnosticare automată.
Logică de control	<ul style="list-style-type: none"> • obținerea de avantaje financiare. • capacitate de prognoză.
Interfața utilizatorului	<ul style="list-style-type: none"> • posibilitatea de a modifica strategiile prestabilite. • caracteristicile inteligente pentru afișare. • disponibilitatea unor diverse tehnologii pentru interfețe sistem-utilizator. • integrarea caracteristicii de monitorizare. • comparația costurilor în funcție de consumul de energie.

Tabelul următor prezintă pașii considerați utili pentru *integrarea consumatorului industrial în rețele inteligente*. În această lucrare se prezintă conceptul în baza căruia se pot implementa etapele prezentate în Tabel 1.3 prin diagrama bloc prezentată în Figura 1.4 și construirea scenariilor pentru control evidențiate în Tabel 1.4. Pe baza acestui concept se bazează prognoza de consum ca element de decizie pentru DR.

Tabel 1.3 Pași pentru implementare DR

PAS 1	Audit energetic complex	<ul style="list-style-type: none"> • raport detaliat despre ce/cum/când poate fi optimizat în așa fel încât să respecte parametrii externi (control, automatizare). • raport detaliat privind disponibilitatea producției proprii (dacă este disponibilă).
PAS 2	Infrastructură de monitorizare	<ul style="list-style-type: none"> • instalarea de contoare inteligente și senzori (ex: termostate). Contoarele inteligente vor determina consumul de energie iar senzorii vor determina cât de eficient este utilizată energia. • centralizarea datelor într-un program de analiză.
PAS 3	Sisteme de control și automatizare	<ul style="list-style-type: none"> • instalarea de relee, electronică de putere, variatoare de viteză, controlul sistemelor de aer condiționat, controlul iluminatului etc.
PAS 4	Infrastructură de telecomunicații în perimetrul consumatorului	<ul style="list-style-type: none"> • activarea comunicațiilor pentru toate procesele în perimetrul consumatorului.
PAS 5	Telecomunicații cu operatorul de sistem	<ul style="list-style-type: none"> • activarea comunicațiilor între consumator și furnizor în timp real și sens bidirecțional.
PAS 6	Crearea scenariilor	<ul style="list-style-type: none"> • analiza, testarea, și crearea scenariilor

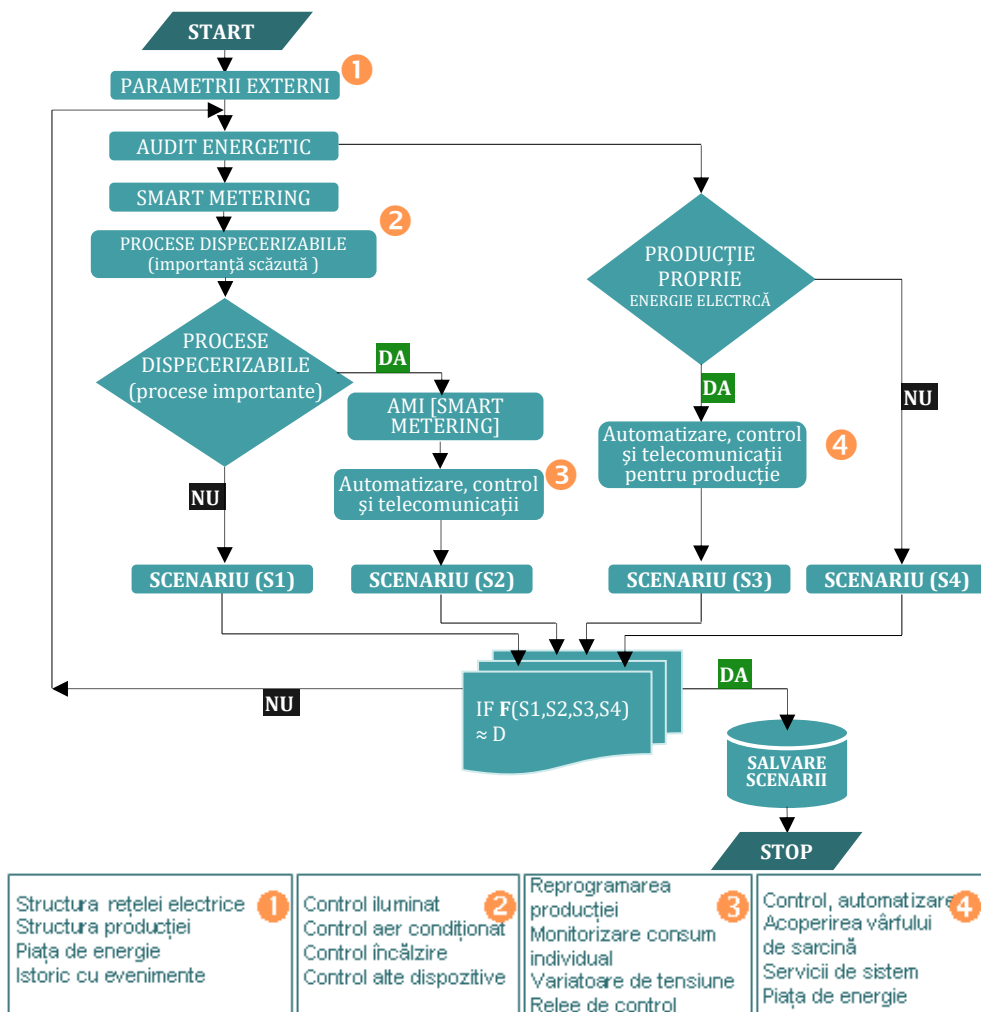


Figura 1.4 Diagramă bloc pentru controlul curbei de sarcină

Această diagramă bloc poate fi pusă în practică prin diferite metode: sisteme expert, rețele neuronale, logică fuzzy sau algoritmi genetici. Scenariile trebuie să îndeplinească toate limitările impuse de procesul tehnologic în așa fel încât procesul tehnologic să nu fie afectat și în același timp să se considere și deciziile companiei. Deoarece parametrii externi se schimbă continuu, iar tehnologia de producție poate suferi schimbări în timp, această diagramă trebuie actualizată constant pentru a oferi soluții optime:

- un scenariu poate avea mai multe strategii.
- scenariile sunt definite și salvate în funcție de eficiența rezultatelor.

- strategiile pot fi implementate la un singur consumator sau la mai mulți consumatori (linii de producție sau secții de producție).
- după determinarea strategiilor posibile, trebuie realizate combinații între scenarii cu scopul de a obține soluția cea mai fiabilă.

Tabel 1.4 Exemplu pentru identificarea scenariilor

SCENARIU S1	SCENARIU S2	SCENARIU S3	SCENARIU S4
Strategie 1 – control iluminat. Strategie 2 – control temperatură (aer condiționat). Strategie 3 – control alte aparate. Strategie 4 – închiderea anumitor secții.	Strategie 1 – control iluminat. Strategie 2 – control secții. Strategie 3 – aplatizarea curbei de sarcină prin reprogramarea proceselor. Strategie 4 – închiderea anumitor secții.	Strategie 1 – control secții. Strategie 2 – acoperirea vârfului de sarcină cu producție proprie. Strategie 3 – tranzacționarea energiei pe piața.	Nu este posibilă producerea de energie electrică în perimetrul consumatorului. Se vor lua în considerare doar scenariile 1 și 2.
S1 = f(STR1,STR2,STR3,STR4)	S2 = f(STR1,STR2,STR3,STR4)	S3 = f(STR1,STR2,STR3,STR4)	S4 = 0

Strategiile de control ale curbelor de sarcină (DSM) trebuie să țină cont de comportamentul particular al fiecărei entități industriale și a tuturor proceselor și echipamentelor din fluxul tehnologic. Curbele de sarcină ale consumatorilor trebuie controlate în materie neinvazivă prin strategii DSM și DR. Experții și managementul companiei trebuie să aprobe sarcinile care urmează să fie reduse și să convină asupra unui program de acțiune și a ghidului de procedură pentru fiecare etapă a declanșării și implementării DSM.

Activitatea principală a unei societăți comerciale este de a obține bunuri și servicii destinate comercializării. Persoanele de conducere din aceste companii nu vor periclita producția datorită unor considerente ecologice sau a unui preț ridicat pe piață, numai dacă există suficiente motive financiare sau legislative. Efortul depus de la începerea unui audit și până la obținerea unor soluții practice de eficiență energetică este considerabil. Multe întreprinderi au propriile lor strategii de management, uneori confidențiale, iar procesele de producție sunt în exploatare de ani de zile și poartă amprenta angajaților. Din acest motiv este foarte greu de aplicat măsuri standardizate, fiecare consumator necesitând o abordare individuală, particulară.

În mod normal, consumatorii industriali au nevoie de o amortizare rapidă a investițiilor pentru tehnologii noi (maxim 3 ani). Investițiile în energie regenerabilă necesită o amortizare mai mare de 3 ani, iar în aceste condiții stimulentele guvernamentale sunt necesare. Dar, uneori, consumatorii industriali au sisteme de rezervă sau procese

care pot fi utilizate pentru producerea de electricitate sau de a reduce consumul (ex: abur fierbinte, apă caldă, gaze fierbinți etc.).

Rezultatul constă în metode fiabile și corecte care interacționează cu rețeaua electrică atât în beneficiul consumatorilor cât și pentru întregul lanț de producere-transport-distribuție a energiei electrice.

1.3. Integrarea consumatorilor medii și mari în smart grid

Controlul curbelor de sarcină în cadrul funcționării SG-ului implică în principal prognoza și gestionarea sarcinilor electrice industriale și comerciale prin definirea unor profile de control care să determine rezerva de capacitate disponibilă pentru DSM și DR la nivel orar. O astfel de abordare implementată cu algoritmi bazați pe învățare automată oferă o mai bună înțelegere la nivel local și regional despre optimizarea software-ului de decizie. Cadrul implementat în acest studiu utilizează măsurători reale de la un grup de consumatori pentru a construi o bază de date cu potențialul DR în România. Cel mai intens cercetat domeniu pentru îmbunătățirea flexibilității sistemelor energetice este managementul cererii (Demand Side Management - DSM), care vizează îmbunătățirea flexibilității prin participarea activă a consumatorilor în menținerea echilibrului producție-consum. Punerea în aplicare a programelor DSM poate varia de la soluții de îmbunătățire a eficienței energetice, spre exemplu folosirea materialelor izolatoare mai performante până la sisteme energetice complet autonome care răspund automat la schimbările din rețea și piața de energie. În lucrare [9] este prezentată o analiză amplă a literaturii de specialitate și o abordare nouă pentru clasificarea metodelor DSM care acoperă toate abordările DSM cunoscute cu o terminologie unificată. Controlul curbei de sarcină (DR) și DSM (Demand Side Management) nu sunt aceleași concepte, chiar dacă sunt adesea utilizate în mod interschimbabil. DSM poate fi implementată în două moduri: prin eficiență energetică sau prin controlul curbei de sarcină (DR). DR se referă la programele care încurajează participanții să facă reduceri pe termen scurt ale consumului de energie. Aceste "răspunsuri" pe termen scurt sunt declanșate de semnalele de preț de pe piața de energie [10] sau inițiate de către operatori de transport (OST) sau operatori de distribuție (DSO). Activările de DR pot dura de la câteva minute până la câteva ore, în funcție de programul prestabilit, și pot include oprirea sau diminuarea iluminatului, ajustarea funcționării

sistemului HVAC sau oprirea unui proces de producție necritic. Sistemele de generare și de stocare aflate în proprietatea consumatorilor pot fi utilizate pentru echilibrarea rețelei electrice. Măsurile de reducere a consumului de energie electrică sunt măsuri temporare, reactive, care mențin funcționarea optimă a rețelei și care atenuează automat vârfurile sau golurile în ceea ce privește cererea și oferta de energie electrică. DSM este orice program care încurajează consumatorul final să fie mai eficient din punct de vedere energetic - astfel încât DR intră în această categorie, dar și măsurile de eficiență energetică pe termen mai lung sau permanent, cum ar fi modernizarea iluminatului, modernizarea automatizării clădirilor și îmbunătățiri ale sistemului HVAC.

Conform [11], este mai solicitant să punem în aplicare DR pentru consumatorii industriali decât consumatorii rezidențiali și clădirile comerciale. Pentru entitățile industriale, DR trebuie să ia în considerare nu numai consumul de energie electrică, ci și fluxul tehnologic al altor resurse, cum ar fi materiile prime, apa și gazul. În al doilea rând, funcționarea unei fabrici este în timp real, iar DR în instalațiile industriale trebuie să îndeplinească cerințe stricte în timp real [12]. În al treilea rând, în instalațiile industriale, întreruperea în alimentare cu energie electrică poate provoca probleme tehnice și financiare. Sectoarele comerciale și industriale se bazează mai mult pe energie electrică pentru încălzire decât sectorul rezidențial care poate avea o dependență mai puternică de gaze naturale, petrol, sau alte surse. Sectorul comercial, în principal în comerțul cu amănuntul al mărfurilor, reprezintă 30-50% din energia electrică utilizată pentru echipamente de refrigerare și depozitare [13]. Refrigerarea este responsabilă de aproximativ 17% din totalul energiei electrice utilizate la nivel mondial și de aproximativ 8% din emisiile de gaze cu efect de seră.

SG vizează accelerarea dezvoltării rețelelor electrice prin creșterea gradului de automatizare și eficientizarea utilizării resurselor de energie [14]. Optimizarea echilibrării sistemului electro-energetic național crește fiabilitatea și calitatea furnizării energiei electrice, împreună cu securitatea rețelei, eficiența energetică și aspectele legate de gestionarea consumului [15]. Sistemele SG se bazează pe infrastructuri avansate de monitorizare care produc o cantitate imensă de date care facilitează analize detaliate și performanțe îmbunătățite ale prognozelor pe tot lanțul energetic (Figura 1.5). În special, prognozarea consumului de electricitate este o sarcină critică în domeniul energiei, deoarece permite un sprijin real pentru luarea deciziilor, susținând

strategii optime de stabilire a prețurilor, integrarea optimă a surselor regenerabile și reducerea costurilor de întreținere.

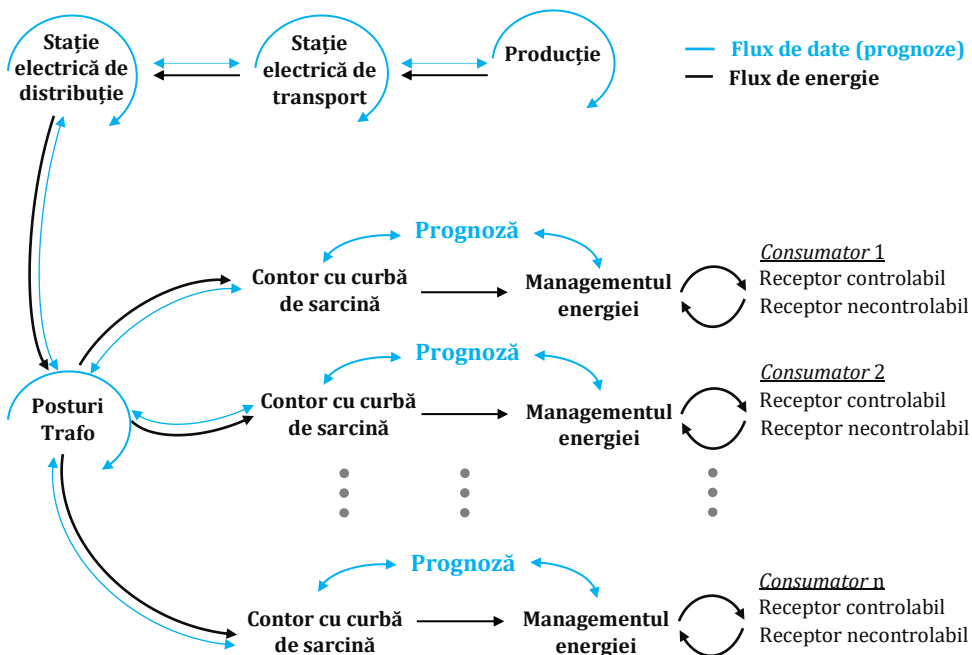


Figura 1.5 Integrarea prognozei în contextul SG

Prognozarea sarcinii se efectuează pe diferite perioade, variind de la secunde la ani, în funcție de problema specifică. Lucrarea de față se va axa pe prognoza de consum non-rezidențial pentru un orizont de timp de 1 oră, 24 ore, 48 ore și 168 ore. S-a ales acest orizont de timp deoarece în practică un furnizor poate folosi o astfel de prognoză să își optimizeze portofoliul de clienți prin participarea la piața de energie electrică. Denumită în literatură ca prognoza consumului pe termen scurt (PTS), acest orizont este compatibil și cu orarul în care pot fi încărcate ofertele de energie pentru PZU (piața pentru ziua următoare) și PI (Piața Intra-zilnică) cât și operarea notificărilor fizice pentru piața de echilibrare (PE) care vor determina costurile de echilibrare pentru un furnizor. Dacă aceste costuri sunt mari, atunci furnizorul va fi obligat să transfere costurile în prețul energiei electrice (având în vedere tarifarea monomă des practică în România). Majoritatea furnizorilor minimizează aceste costuri prin agregarea curbelor de consum de la mai mulți consumatori, reușind astfel o reducere comercială a costului echilibrării consumului,

dar fără să ajute din punct de vedere tehnic rețeaua electrică care trebuie să suporte fluctuațiile consumului.

Liberalizarea pieței de energie electrică și adoptarea pe scară largă a surselor regenerabile de energie influențează puternic prețurile pieței, iar PTS reprezintă un instrument important pentru creșterea competiției și calității furnizării energiei electrice. Începând cu 2021 o modificare legislativă a impus schimbarea intervalului de decontare de la o oră la 15 minute, ceea ce determină costuri mai mari cu dezechilibrele. Costuri mai mari, deoarece furnizorii trebuie să prognozeze următoarea zi pentru 96 de pași, ceea ce din punct de vedere tehnic reprezintă o mare provocare.

În scenariul unui singur consumator, prognoza sarcinii este destul de dificilă, deoarece seriile de timp aferente consumului de energie sunt extrem de variabile. În scenariul în care sarcina este agregată, respectiv consumul asociat cu mai multe tipuri de consumatori industriali, modelul de consum este în mod normal mai ușor de prezis deoarece variația curbei de sarcină urmează o dinamică mai lentă. Scopul tezei este de a dezvolta metode care “învață” pe seturi de date și sunt apoi folosite în mod repetitiv pentru realizarea prognozelor. Având în vedere modalitatea de obținere al datelor de la operatorul de distribuție la finalul fiecărei luni, metodele de prognoză nu pot fi actualizate între prognoze. În contextul pieței de energie, date cât mai actuale sunt absolut necesare pentru realizarea prognozelor. Datele actualizate pot fi utilizate pentru a crea o nouă metodă sau pentru actualizarea metodei existente înainte de fiecare prognoză devenind un model dinamic. De exemplu, dacă este necesară o prognoză la începutul săptămânii s_1 pentru săptămâna următoare s_2 , se pot folosi valorile reale la sfârșitul săptămânii s_1 pentru actualizarea modelul înainte de a face prognoza săptămânii următoare. Acesta ar fi un model dinamic. Dacă nu se obține o observație reală la sfârșitul săptămânii sau se decide să nu se adapteze modelul de prognoză, acesta ar fi un model static. Constrângerile infrastructurii de monitorizare impune limitarea la o abordare statică. În prezenta lucrare sunt folosite curbe de sarcină aferente consumatorilor industriali și comerciali cu activitate în domeniul prelucrării lemnului și producției de mobilier, în domeniul comerțului cu amănuntul și alimentar. În Tabel 1.5 se prezintă consumul final de energie electrică în funcție de activitatea industrială la nivelul anului 2019 și se evidențiază impactul domeniilor analizate în această lucrare în mixul național de consum.

Tabel 1.5 Consumul final de energie electrică pe activități industriale în anul 2019⁴

	GWh	% din
Total	21.678,7	100,00
Extracția minereurilor metalifere	119,75	0,55%
Alte activități extractive	132,25	0,61%
Alimentară, băuturi și tutun	1.893,25	8,73%
Textile și alte produse textile	264	1,22%
Confecții din textile, blanuri și piele	231,25	1,07%
Pielarie și încălțăminte	184	0,85%
Prelucrarea lemnului	1.002,75	4,63%
Celuloză, hârtie și carton	557,25	2,57%
Edituri, poligrafie și reproducerea înregistrărilor pe suport	162,25	0,75%
Chimie și fibre sintetice și artificiale	2.006,25	9,25%
Prelucrarea cauciucului și maselor plastice	1.129,25	5,21%
Alte produse din minerale nemetalice	2.589,25	11,94%
Metalurgie	5770	26,62%
Construcții metalice, mașini și echipamente	4.079,0	18,82%
Mobilier și alte activități neclasificate	539,75	2,49%
Captarea, tratarea și distribuirea apei	593,75	2,74%
Construcții	425,5	1,96%

În Figura 1.6 se poate observa impactul sectorului industrial și al serviciilor care reprezintă cel mai mare consum de energie electrică la nivel național. În perioada anterioară anului 1990, dezvoltarea economiei românești a fost bazată pe dezvoltarea ramurilor energointensive ale industriei grele. Restructurarea sau tranziția economiei către descentralizare în toate sectoarele economice a dus la o scădere majoră a consumului final energetic în sectorul industrial. Aceste efecte au fost intensificate de criza economică, astfel încât, după anul 2010, sectorul industrial nu a mai ocupat poziția de lider în ceea ce privește ponderea în consumul energetic final, locul său fiind preluat de sectorul casnic din cauza recesiunii din anul 2009.

⁴ <http://statistici.insse.ro:8077/tempo-online/#/pages/tables/insse-table>

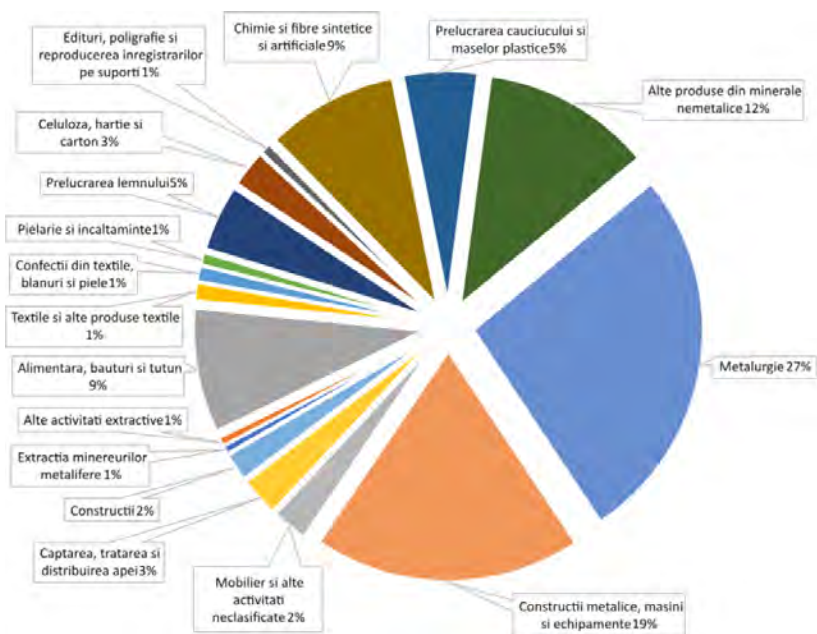


Figura 1.6 Consumul final de energie electrică în industrie (2019)⁵

În anul 2016, consumul final energetic în sectorul industrial a continuat să scadă. Sectorul și-a păstrat poziția de al doilea mare sector consumator de energie din România, înregistrând un consum de energie de 6,3 Mtep, ceea ce a reprezentat 28,24% din consumul final energetic la nivel național⁵ (Figura 1.7). În ceea ce privește ponderea în consumul de energie electrică la nivel național, în anul 2016, sectorul industrial este principalul consumator cu o pondere de 48 % (Figura 1.8).

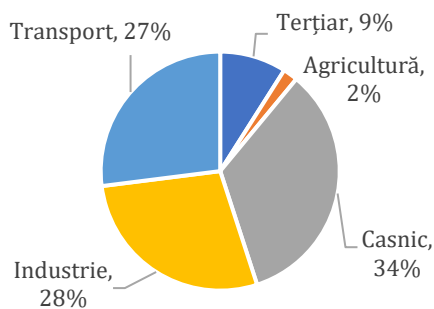


Figura 1.7 Ponderea consumului de energie în totalul de energie – 2016⁵

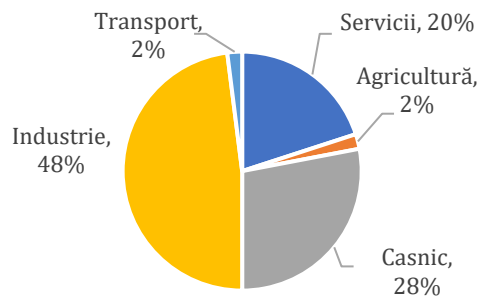


Figura 1.8 Ponderea consumului de electricitate în industrie – 2016⁵

⁵ <https://www.anre.ro/ro/eficienta-energetica/rapoarte/rapoarte-activitate>

În perioada 2012-2016, evoluția consumului de energie în sectorul industrial a fost descendentă, identică cu evoluția consumului final total. Valoarea acestui consum a scăzut cu 495 ktep, adică cu 7,28% și cu o depreciere de 1,45 %/an. Ponderea acestuia în consumul total de energie a scăzut cu 2%, de la 30% în 2012 la 28% în 2016. Această reducere a consumului poate fi observată și în Figura 1.9 prin evidențierea emisiilor de CO₂ care au scăzut direct proporțional cu scăderea consumului industrial.

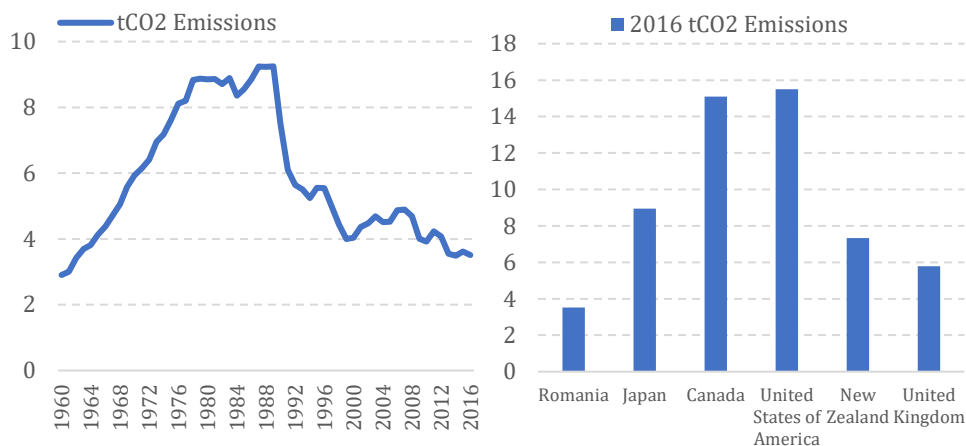


Figura 1.9 Impactul consumului de energie asupra emisiilor de CO₂ ⁶

2. Prognoza consumului de electricitate

Tipurile de prognoză pentru consumul de electricitate se împart în trei categorii principale după cum este prezentat în Tabel 2.1, în funcție de orizontul de timp pentru care se dorește prognoza. Autorii din [16] prezintă o clasificare bazată pe isotircul de date utilizat înainte de prognoză, incluzând noțiunea de prognoză pe termen foarte scurt și un cadru de clasificare bazat pe intervalul prognozat. În acest studiu ne raportăm la terminologia care se referă la prognoza pe termen scurt, deoarece prognozele se fac pentru 1, 24, 48, 168 de pași (ore). Această abordare permite aplicabilitatea prognozelor în piața de energie electrică conform produselor specifice de tranzacționare.

⁶ <https://datacatalog.worldbank.org/dataset/world-development-indicators>

Tabel 2.1 Tipuri de prognoză pentru de consum de electricitate

Tip	Orizont	Aplicabilitate
Prognoză pe termen scurt.	De la secunde la ore sau săptămâni.	Controlul circulațiilor de puteri; Echilibrare economică și angajamentul optim al unităților de producție; Controlul în timp real și evaluarea securității rețelelor; Licitatii pentru piata energiei.
Prognoză pe termen mediu.	De la săptămâni până la ani.	Programe de întreținere; Coordonarea capacităților dispecerizabile; Decontarea prețurilor pentru serviciile auxiliare.
Prognoză pe termen lung.	De la câțiva ani până la 10-20 ani.	Planificarea extinderii sistemului pe termen lung; Legislație; Strategii Energetice.

Rezultatele diverselor metode și modele implementate pentru prognoză pe termen scurt (PTS) variază în funcție de mulți factori. Principalul factor este tipul consumatorului prognozat. Există trei categorii principale de consumatori în conformitate cu [17], iar în [18] autorii exemplifică patru macro-categorii în funcție de tipul de consumator: industrial, comercial, rezidențial și autorități publice. Curbele de sarcină ale fiecărei categorii sunt serii orare temporale care variază în funcție de mulți factori. Consumatorii industriali sunt cei mai greu de prognozat din cauza multitudinii de dispozitive și echipamente utilizate de angajați pe parcursul întregii zile de lucru. Curbele de sarcină industriale sunt mai puțin deterministe, iar modelele zilnice sau orare au variații ridicate. În ciuda dezvoltării metodelor de prognoză, niciuna nu poate fi generalizată pentru toate tipurile de consum [19]. În cazul consumatorilor comerciali sau rezidențiali, metodele de prognoză utilizează cu succes variabile exogene, cum ar fi temperatura, umiditatea, radiația solară sau precipitațiile. În lucrarea [20] se prezintă impactul variabilelor exogene în prognoza sarcinii, iar [21] subliniază faptul că utilizarea temperaturii ca variabilă exogenă nu îmbunătățește acuratețea predicției din zilele lucrătoare pentru consumatorii casnici; rezultate pragmatice sunt obținute folosind temperatura doar pentru ziua de duminică.

O mulțime de abordări au fost aplicate pentru prognoza consumului de electricitate, constând din analiza seriilor de timp, regresie, tehnici de netezire, inteligență artificială, rețele neuronale artificiale și diverse metode hibride. Toate aceste metode pot face acest domeniu de cercetare copleșitor. Unii autori sugerează că modelele consacrate funcționează mai bine, analizele din lucrările [22], [23] și [24] prezintă dovezi care indică efectul negativ al complexității în precizia prognozării. Autorii în [25] propun "*Regula de aur*" pentru a se ajunge la o teorie unificată a prognozei, în timp ce alți cercetători încorporează mai mulți algoritmi pentru a construi metode hibride care combină caracteristicile statisticilor tradiționale și învățare automată.

Există argumente solide de ambele părți; anumiți algoritmi vor furniza rezultate mai bune sau mai rele în funcție de datele istorice sau orizontul de timp prognozat. Obiectivele prognozelor sunt reducerea la minimum a erorilor și îmbunătățirea activității economice a furnizorului și consumatorului. Prognozele aduc valoare dacă sunt implementate de consumatori în colaborare cu furnizorii de energie pentru optimizarea proceselor prezentate în Figura 2.1. Pentru a veni în sprijinul consumatorilor se urmărește utilizarea metodelor de învățare automată pentru PTS (Prognoza pe Termen Scurt), astfel încât clientul să fie sprijinit în această activitate de algoritmi care pot identifica automat evoluția consumului.

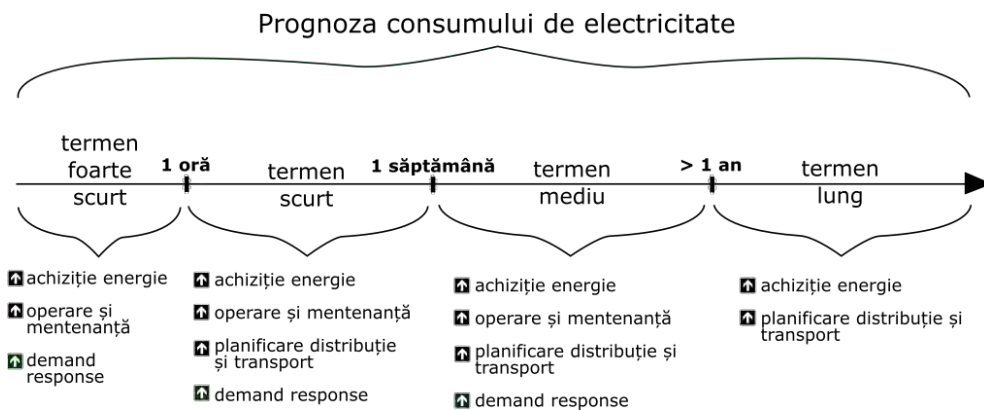


Figura 2.1 Aplicabilitatea prognozei de consum de electricitate[26]

Organizarea concursurilor de prognoză, în genul celor prezentate în [27] și [28] reprezintă una dintre cele mai bune modalități de a compara algoritmi pe date istorice similare și de a indica rezultatele. În mai multe cazuri, arhitecturile bazate pe rețele neuronale recurente sunt prezentate ca un algoritm stabil cu rezultate bune. Autorii în lucrarea [29] au efectuat un studiu experimental amplu folosind șapte arhitecturi populare cu învățare adâncă (DL) și au constatat că algoritmul cu memorie pe termen lung-scurt (LSTM) este cel mai robust tip de rețea recurentă și oferă cea mai bună precizie de prognozare, iar rețelele neuronale convoluționale (CNN) obțin rezultate inferioare dar au o variabilitate mai mică a rezultatelor. Literatura de specialitate indică faptul că seriile temporale ale consumului de electricitate pot fi simultan liniare, neliniare, sezoniere, nesezoniere și incerte. Metodele pentru prognoză trebuie să fie capabile să gestioneze serii de timp pentru toate tipurile și caracteristicile de consum.

2.1. Abordarea tradițională

În literatură, clasificarea prognozei determină două categorii principale ilustrate în Figura 2.2: calitativă și cantitativă. Studiul detaliat de autorii lucrărilor [30] și [31] prezintă o revizuire cuprinzătoare și detaliată a selectării metodelor de prognoză din punct de vedere econometric. În ceea ce privește analiza curbele de sarcină electrică, prognozarea are aceleași principii fundamentale, cu puține particularități. Tehnicile calitative aplică cunoștințe empirice ale experților pentru a obține prognoze. Tehnicile calitative utilizate pentru predicția pe termen mediu-lung sunt în primul rând opinii ale experților, cum ar fi Metoda Delphi, Cercetarea pieței sau Consensul Experților prezentate de [32].

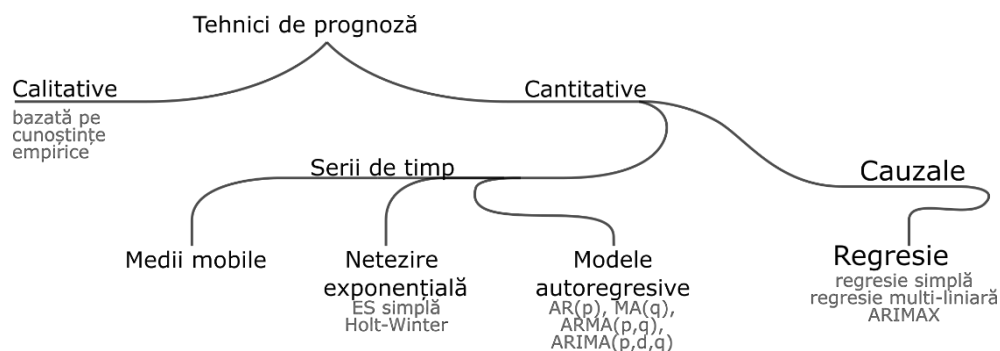


Figura 2.2 Tipuri de prognoza : calitative și cantitative

Tehnicile cantitative sunt eficiente pentru prognozele pe termen scurt și constau din i) prognoza seriilor temporale și ii) prognoza cauzală. În prognoza seriilor de timp, datele istorice sunt un set de puncte de observație ordonate cronologic unde X_t este înregistrat în timp, respectiv valorile observate la momente $t = 1, 2, \dots, n$ (X_1, X_2, \dots, X_n). În contrast cu prognoza cauzală, prognoza seriilor de timp (PTS) este o estimare ordonată a punctelor de date măsurate. PTS ajută la înțelegerea componentelor precum tipare, tendințe, valori de vârf și orice neregularitate sau variație a seriei de date. În seriile de timp, pot fi aplicate două forme de descompunere: descompunerea aditivă și descompunerea multiplicativă: $x_t = m_t + s_t + e_t$ și $x_t = m_t \cdot s_t \cdot e_t$, unde m_t este tendința, s_t este sezonalitatea și e_t este eroarea sau zgomot aleatoriu.

Media mobilă (MM) sau media mobilă simplă este cea mai simplă modalitate de a prognoza calculând o medie a ultimelor k perioade evidențiată în relația (2.1) Valoarea medie este considerată a fi valoarea prognozată pentru perioada următoare.

$$\hat{Y}_{n+1} = \frac{X_{n-k+1} + X_{n-k+2} + \dots + X_n}{k} = \frac{\sum_{i=n-k+1}^n X_i}{k} \quad (2.1)$$

Netezirea exponențială (NE) este o tehnică utilizată frecvent, în care o serie de timp "netezită" este generată prin atribuirea unor ponderi variabile în funcție de istoricul datelor. Metoda NE este potrivită pentru seturi de date fără trend și cu niveluri variabile. Metoda de netezire exponențială simplă introdusă de [33] este prezentată în relația 2.2 unde se poate observa faptul că se ține cont de previziunea pentru perioada anterioară și se ajustează cu eroarea de prognoză. Prognoza pentru următoarea perioadă devine:

$$\hat{Y}_{t+1} = \hat{Y}_t + \alpha(y_t - \hat{y}_t) \quad (2.2)$$

unde t = perioada de timp curentă; \hat{Y}_{t+1} = prognoză pentru perioada următoare; y_t = date reale pentru perioada curentă; \hat{y}_t = prognoză pentru perioada curentă realizată în ultima perioadă; ($0 < \alpha < 1$) constantă de netezire. În funcție de α , noua prognoză se poate schimba semnificativ față de cea veche. Prin urmare, putem scrie actualizarea continuând această extindere:

$$\hat{y}_{t+1} = \alpha y_t + \alpha(1 - \alpha) Y_{t-1} + \alpha(1 - \alpha)^2 y_{t-2} + \alpha(1 - \alpha)^3 y_{t-3} + \dots + \alpha(1 - \alpha)^{t-1} y_1 + \alpha(1 - \alpha)^t y_1 + \dots \quad (2.3)$$

Îmbunătățirile aduse la metoda NE simplă sunt metodele lui Holt și Winter aplicate pentru seturi de date cu trend variabil. Propusă de Holt (1957)[34] și studiată de Winters (1960)[35], metoda care ulterior a fost denumită Holt-Winters (HW) constă din trei ecuații de netezire pentru nivel, trend și sezonaliitate. Holt a extins NE simplă la netezirea exponențială liniară, permițând prognozele cu trend. Configurarea parametrilor de prognoză prin două constante de netezire α și β cu valori între (0..1) și prin rescrierea $\ell_t = \hat{y}_{t+1}$ rezultă următoarele ecuații:

$$\text{Nivel:} \quad \ell_t = \alpha y_t + (1 - \alpha)(\ell_{t-1} + b_{t-1}) \quad (2.4)$$

$$\text{Creștere:} \quad b_t = \beta(\ell_t - \ell_{t-1}) + (1 - \beta)b_{t-1} \quad (2.5)$$

$$\text{Prognoză:} \quad \hat{Y}_t = (\ell_{t-1} + b_{t-1}) \quad (2.6)$$

Unde ℓ_t este estimarea nivelului la momentul t , și b_t este o estimare a pantei (sau creșterii) la același moment t . Metodele HW sunt o extensie a metodei liniare Holt, dar metodele HW diferă în funcție de descompunerea sezonieră care poate fi aditivă sau multiplicativă.

$$\text{Nivel:} \quad \ell_t = \alpha \frac{y_t}{s_{t-m}} + (1 - \alpha)(\ell_{t-1} + b_{t-1}) \quad (2.7)$$

$$\text{Creștere:} \quad b_t = \beta(\ell_t - \ell_{t-1}) + (1 - \beta)b_{t-1} \quad (2.8)$$

$$\text{Sezonier:} \quad s_t = \gamma \frac{y_t}{\ell_{t-1} + b_{t-1}} + (1 - \gamma)s_{t-m} \quad (2.9)$$

$$\text{Prognoză } t+1: \quad \hat{Y}_{t+1} = (\ell_t + b_t)s_{t-m+1} \quad (2.10)$$

$$\text{Prognoză } t+k: \quad \hat{Y}_{t+k} = (\ell_t + k \times b_t)s_{t-m+k} \quad (2.11)$$

unde m este perioada sezonieră, ℓ_t , b_t au fost definiți anterior, s_t este componenta sezonieră. Parametrii (α , β și γ) sunt de obicei limitați să se situeze între 0 și 1.

O abordare simplistă pentru prognozarea seriilor de timp este implementarea metodelor naive care reprezintă un etalon util pentru compararea prognozelor, deoarece sunt des folosite în practică. Pentru prognoze naive, pur și simplu se stabilesc toate prognozele în funcție de valorile ultimelor observații disponibile, $\hat{y}_t = y_{t-1}$. Autorii din [36] au implementat netezirea exponențială Holt-Winters în comparație cu două metode naive de referință care arată inferioritatea acestora din urmă. Diverse ecuații [37] pot fi utilizate pentru a implementa astfel de algoritmi, de exemplu, metoda medie:

$$\hat{Y}_t = \bar{y} = \frac{(y_1 + \dots + y_{t-1})}{t - 1} \quad (2.12)$$

O metodă care ține cont de componenta ciclică poate fi scrisă conform ecuației (2.12). Prognoza pentru fiecare interval (oră) se realizează prin utilizarea valorilor din aceeași perioadă a zilei sau

săptămânii anterioare (de exemplu, în aceeași zi a săptămânii precedente). Prognoza pentru ora $t + h$ este scrisă sub forma:

$$\hat{Y}_{t+h} = Y_{t-m(k+1)} \quad (2.13)$$

unde m reprezintă perioada ciclică, și k este numărul de pași în perioada de prognoză anterioară timpului t .

O metodă naivă similară poate fi implementată prin creșterea sau scăderea unui factor în timp (sens), care poate fi o modificare medie a valorilor folosite în prognoză ($y_1 + \dots + y_t$). De exemplu, prognoza pentru timpul $T + h$ este dată de:

$$\hat{Y}_{t+h} = y_t + \frac{h}{t-1} \sum_{t=2}^t (y_t - y_{t-1}) = y_t + h \left(\frac{y_t - y_1}{t-1} \right) \quad (2.14)$$

Metoda bazată pe autoregresie diferențiată și medie mobilă (ARIMA) este o metodă statistică care interpretează seriile de timp pentru a prezice valorile viitoare pe baza corelațiilor din trecut. Modelul ARIMA a fost propus de Box & Jenkins [38] și are trei componente: autoregresiv, diferențiere și media mobilă aplicate setului de date. ARIMA corelează în mod esențial abaterile sale anterioare de la medie. Identifică trend, sezonaliitate, cicluri, erori și aspecte non-staționare ale unui set de date atunci când realizează prognoze folosind relațiile (2.15)-(2.20). Aceste modele se bucură de o largă popularitate datorită rezultatelor de înaltă precizie, a flexibilității și a modelelor matematice riguroase pentru prezicerea seriilor temporale [39].

$$\text{AR}(p): \quad \hat{y}_t = \Phi_1 Y_{t-1} + \dots + \Phi_p Y_{t-p} + \varepsilon_t \quad (2.15)$$

$$\text{MA}(q): \quad \hat{y}_t = \varepsilon_t + \beta_1 \varepsilon_{t-1} + \dots + \beta_q \varepsilon_{t-q} \quad (2.16)$$

$$\text{ARMA}(p,q): \quad \hat{y}_t = \Phi_1 Y_{t-1} + \dots + \Phi_p Y_{t-p} + \varepsilon_t + \beta_1 \varepsilon_{t-1} + \dots + \beta_q \varepsilon_{t-q} \quad (2.17)$$

$$\text{Operator de întârziere:} \quad L^k y_t = y_{t-k} \quad (2.18)$$

$$\text{Operator de diferențiere:} \quad \Delta^2 Y_t = (1 - L)^2 y_t \quad (2.19)$$

$$\text{ARIMA}(p,d,q): \quad \left(1 + \sum_{k=1}^p \alpha_k L^k \right) (1 - L)^d Y_t = \left(1 + \sum_{k=1}^q \beta_k L^k \right) \varepsilon_t \quad (2.20)$$

Metoda cauzală de prognozare este o tehnică care determină corelația dintre valori care trebuie prezise și alte variabile exogene. Această metodă ar trebui să ia în considerare toate posibilele variabile independente care influențează evoluția variabilei dependente. Se determină o funcție care prognozează variabila dependentă (Y) cu variabilele independente (X_1, X_2, \dots, X_k) [40].

Regresie liniară simplă:
$$\hat{y}_t = \beta_0 + \beta_1 X_t + \varepsilon_t \quad (2.21)$$

Regresie multiliniară:
$$\hat{y}_t = \beta_0 + \beta_1 X_{1,t} + \beta_2 X_{2,t} + \dots + \beta_k X_{k,t} + \varepsilon_t \quad (2.22)$$

unde \hat{Y} sunt valorile variabilei prognozate și (X_1, X_2, \dots, X_k) sunt variabilele „ k ” exogene. Coeficienții β_1, \dots, β_k din ecuația 2.21 cuantifică corelația fiecărei variabile exogene cu variabila dependentă. Modelele cauzale de prognoză sunt sofisticate și complexe, deoarece utilizează informații specifice despre relațiile în timp dintre variabilele care influențează evoluția consumului, cum ar fi temperatura, prețul electricității, vremea și alți factori socio-economici. La fel ca în cazul analizelor de serii temporale, datele istorice sunt esențiale pentru crearea unei prognoze cauzale.

2.2. Învățare automată (machine learning)

Există un vast spectru terminologic care tinde să fie confuz din cauza utilizării interschimbabile a conceptelor: inteligență artificială, învățare automată, învățare profundă, rețele neuronale artificiale sau învățare consolidată. Diferența dintre o rețea neuronală simplă și un algoritm de învățare adânc este numărul de neuroni (Figura 2.3), performanța și structura straturilor ascunse. Pentru ca o rețea neuronală să fie considerată cu învățare adâncă trebuie să utilizăm mai mult de două straturi ascunse. Învățarea automată este considerată un subdomeniu al inteligenței artificiale [41]. Învățarea adâncă este un subdomeniu al învățării automate după cum este prezentat în Figura 2.4.

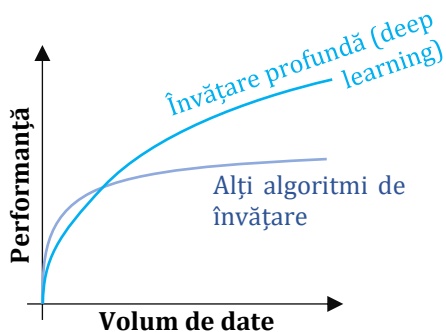


Figura 2.3 Legătura dintre volumul de date și performanță⁷[42]

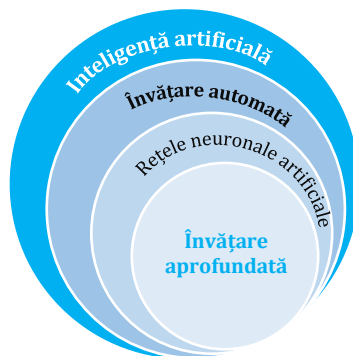


Figura 2.4 Domeniile inteligenței artificiale⁷

Tehnicile bazate pe învățare automată prezentate în Figura 2.5 pot fi grupate în două grupe mari: învățare supravegheată și nesupravegheată. Metodele care utilizează învățarea supravegheată se bazează pe configurarea parametrilor specifici utilizând setul de date de antrenare împărțit în date de intrare și valori de ieșire cunoscute (țintă).

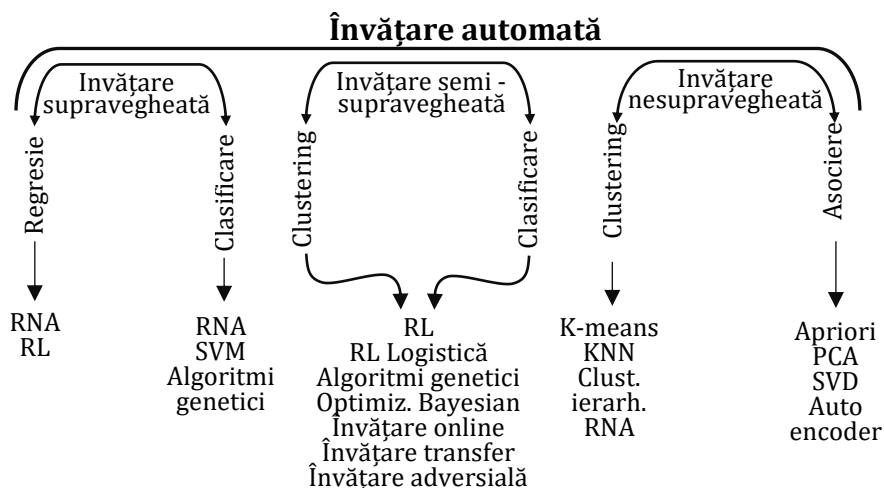


Figura 2.5 Clasificarea algoritmilor de învățare automată[43]

Tehnicile de învățare nesupravegheate formează grupuri între obiectele dintr-un lot identificând asemănările și apoi le folosesc pentru identificarea necunoscutelor. Învățare consolidată (reinforcement learning) este un algoritm comportamental similar cu învățarea

⁷ <https://www.slideshare.net/ExtractConf>

supravegheată, dar nu utilizează date eșantion pentru antrenament, ci stabilește aleator ce date sunt folosite în procesul de învățare. O succesiune de rezultate de succes va dezvolta cea mai bună recomandare sau metodă pentru o anumită problemă. Mai multe articole precum [44] prezintă o revizuire generală a metodelor de inteligență artificială iar lucrările [26], [45] și [46] detaliază o revizuire cuprinzătoare pentru algoritmi de învățare automată utilizați pentru prognozarea curbelor de sarcini. În lucrarea [47] autorii folosesc învățarea adâncă pentru prognozele probabilistice multivariate. Prognoza pe termen scurt a seriilor de timp în literatura de specialitate se consideră a avea un orizont de prognoză mai mare de 1 oră [48]. Pentru prognozarea consumului industrial în [49] s-a implementat un model folosind o metodă de descompunere adaptivă pentru a procesa modelul de regresie pe date brute pentru fiecare serie de timp utilizată. Modele statistice simple pot fi eficiente, așa cum autorii au prezentat în [50] o metodă bazată pe o medie mobilă de 7 zile pentru consum regional și s-a obținut o eroare medie de 30,55 MW și MAPE de 3,84%. Au fost studiate diverse rețele neuronale, spre exemplu în [51] autorii indică utilizarea rețelelor recurente ca fiind cea mai bună metodă de prognoză cu MAPE de 1,55% și [52] prezintă învățarea supravegheată utilizată pentru prognoza pe 24 de ore a consumului de electricitate. În [53] s-au analizat metodele de prognozare a curbelor de sarcină bazate pe 126 de lucrări de cercetare ale prognozei pe termen scurt. În [54] se prezintă o comparație între ML și metode tradiționale (SARIMAX). Mașinile vectoriale de sprijin (SVR) și rețele neuronale au fost aplicate de [55] pentru prognoza consumului clădirilor (campus universitar) și au obținut o eroare medie de 3,46-10 %, iar rețelele neuronale au avut un rezultat mai bun decât SVR. În Tabel 2.2 se prezintă o sinteză a studiului literaturii de specialitate prin care se facilitează comparația prezentei teze cu rezultatele din literatura având în vedere resursele, tipul de curbe de sarcină, orizontul de prognoză și rezultate.

Tabel 2.2 Evidențierea studiilor analizate în literatură

Tip sarcină	Resurse Soft / Hard	Metodă utilizată	Orizont de prognoză	Rezultate [MAPE %]	Ref.
Nivel OD	Matlab, TF/ Intel Core i5 6500 CPU @3.20 GHz, 8 GB RAM	Algoritm hibrid (FA) bazat pe SVM (suport vector machines), RF și LSTM pentru îmbunătățirea prognozei.	Curbă de sarcină la interval de 15-min (5760 înregistrări) Antrenare pe 5664 înregistrări; Test pe 96 înregistrări	SVM=23,4 %; RF=3,6 %; LSTM=5,3 %; FA=2,8 %;	[56]

Clădiri agregate și cartiere	Py, Scikit, Keras/ Intel i7 2.5 GHz 16GB RAM GF 940M	MLP, LSTM, Seq2Seq, RF, KNN. Metoda RF obține cele mai mici erori.	15 minute 2 ore 24 ore	15 min RF = 1,67 ± 1,58 %; 2 hours RF = 2,83 ± 2,73%; 24 hours RF = 4,80 ± 4,24%	[57]
Sarcină industr.	-	Algoritm hibrid CNN (RICNN) care combină RNN cu 1-D CNN (1-DCNN). Comparație cu MLP, RNN, and 1D CNN	24 ore (48 de valori pentru 30 min. Antrenare (75%) și test (25%).	RICNN= 4,71 - 8,88 %	[58]
Sarcină industr.	-	Algoritm hibrid GA (algoritmi genetici) + PSO (particle swarm optimization) + NN. Comparație cu GA BPNN și PSO BPNN (BPNN), PSO, GA	Antrenare = 59 zile Test = 1 zi	GA PSO BPNN = 0,77%; PSO BPNN = 0,95%; GA BPNN = 1,88%	[59]
Nivel OD	Keras; TF; Python/ Intel i7 Nvidia GF RTX™2070	Metodă hibridă între CNN, bi-LSTM și GRU 2-D CNN LSTM GRU	Ora următoare	2-D CNN LSTM GRU = 0,362%	[60]
Sarcină industr.	MATLAB™ R2016b	Metodă sezonieră gri AWBO (average weakening buffer operators) - DGGM(1,1), GM(1,1), DGGM(1,1) și SVM	Date trimestriale din 2013 până în 2018, Q1- 2019 până în Q1-2020 pentru prognoză	2AWBO-DGGM (1,1): 6,79 % GM(1,1): 19,69 % DGGM(1,1): 8,89% SVM: 15,45% 1AWBO-DGGM (1,1): 7,67%	[61]
Nivel național	MATLAB/ intel core i7 2.5 GHz and 8 GB Ram	Metodă cu feedback în care predicția orară printr-o extindere a seriei Fourier este actualizată utilizând eroarea la ora curentă pentru următoarea oră.	(2012–2017) ora următoare; ziua următoare	1 h : 0,87 % 1 zi : 2,90% 1 an : 3,54%	[62]
Nivel OD	-	Algoritm hibrid: S-ARIMA, S-ESM and W-SVM. Ponderile pentru fiecare model au fost determinate cu APSO (adaptive particle swarm optimization)	Ziua următoare pentru 48 de pași (30 min) pentru 5săptămâni	S-ARIMA=1,84-5,38 W-SVM = 2,56 - 7,2 S-ESM = 2,08 - 7,49 Hibrid = 1,28 - 4,36	[63]
Nivel național	EvIEWS pachet software	(F)(S)ARIMA; (S)MLP; DE; GA; PSO; ANFIS; (S)ARIMA/ (S)MLP; Metodă hibridă (PCLR) Bazată pe FSARIMA și FSMLP	Ora următoare 6 săptămâni istoric (6048 înregistrări la 10 min). Antrenare= 5 sapt. Test = ultima săpt.	Metodă propusă = 0,229 %; Restul de metode variază de la 0,231 - 0,389 %	[64]
utility/ municipality level	-	Metodă hibridă bazată pe metoda empirică de descompunere (IEMD), ARIMA, NN și algoritm fruit fly (FOA)	1 h, 1 zi, 1 săpt. 1 lună	Media valorilor MAPE pentru metoda hibridă: 0,66 %	[65]
Nivel OD (doua regiuni)	MATLAB R2017a/	Algoritm nou multi-obiectiv (MOFTL) bazat pe algoritmul <i>Follow the Leader</i> și NN. Comparație cu NSGA-II-	ISO: Antrenare date orare 2004 - 2007, Test 2008 -2009. Ercot:	ISO MOFTL reduce mape cu 17,42%, 6,81%, 10,77% și 59,69%	[66]

	Intel i5 CPU 2.80GHz cu 1 procesor	ANN, FTL-ANN, BPNN, GRNN	Antrenare date orare: 2009 - 2014, Test: 2015, 2016	ERCOT MOFTL reduce mape cu 4,20%, 4,16%, 1,14% și 21,85%	
Nivel consum oraș	-	Metodă hibridă CNN extrage caracteristici din curbele de sarcină și rețele recurente (LSTM) cu straturi dense	Ora următoare Istoric orar 1017 zile (19,608 ore antrenare, 2400 validare și 2400 test	Regresie = 2,93% SVR= 1,72% DNN = 1,66 % CNN RNN= 1,47% CNN RNN Paralel = 1,40 %	[67]
Consum casnic	AMD Ryzen 4.20 GHz 128 GB RAM, 4x NVIDIA GeForce RTX 2080 Ti 11 GB	Metoda RNN adaptiv[online, comparată cu offline LSTM și regresie online MLP, regresie liniară, KNN.	Istoric orar pentru 3 ani 25.559 valori 1 h, 50 h, 100 h, 200 h 70% antrenare 30% test	1h MAE Off. lstm: 0,38 - 11,88; On. Adapt. rnn: 0,071 - 0,38; 50h MAE Off. lstm: 0,43 - 14,26; On. Adapt. rnn = 0,05 - 0,53	[68]
Clădiri birouri	PyTorch 2GPU GF RTX 2080 Ti și 1GPU GFGTX 1060	Rețele Recurente secvențiale (S2S RNN) cu Mecanism Atenție. Comparăție cu RNN simplu, LSTM, și GRU.	Orizont prognoză: 1 oră, 4 ore, 10 ore, 24 ore. 15 luni istoric cu înregistrări la 5 min.	S2S-BA = 1 oră : 2,705 % 4 ore : 4,116% 10 ore : 5,599% 24 hours : 6,774%	[69]
Serii de timp diverse - concurs prognoz.	TF 1.12/ Intel Core i7, 12cores 65Gb,GF GTX 1080 Ti Intel Xeon Gold, 50GB, nVidia Tesla V100; Intel Xeon CPU E5	Diverse arhitecturi RNN. Comparății cu ETS și ARIMA arată că modelele semi-automate RNN aplicabile în orice condiții, dar sunt totuși alternative competitive în multe situații.	CIF 2016 = 6, 12 pași NN5 = 56; M3= 18; M4=18; Wikipedia Web Traffic (Kaggle) = 59; Tourism = 24	CIF : 10,23 % (S2SD LSTM); Kaggle = 45,62% (Stacked GRU adam); M3: 14,11% (S2SD GRU adagrad); NN5 = 21,53% (Stacked LSTM adam); Tourism = 19,02 % (ETS); M4 = 13,08% (auto.arima)	[70]
Cladiri de birouri	-	Hybrids of RNN and clustering. lstm; lstm with attention, BiLSTM, BiLSTM with attention; cnn lstm; cnn Bilstm; Conv2D lstm; Conv2D+Bilstm; Encoder-Decoder	1 oră 24 ore	(24-h) Bldg2 LSTM = 22,37; Bldg3 LSTM = 11,37; Bldg4 LSTM w/ATTN = 5,96; Bldg5 LSTM = 2,25	[71]
Consum casnic	-	Rețele artificiale Elman (sau rețele recurente simple RNN)	1 oră orizont Istoric 79 săptămâni consum casnic orar	Minnum Mape = 1,5% Maximum Mape = 4,6%	[72]
Consum casnic	MATLAB / Laptop PC with 2.40 GHz Intel i3 6 GB RAM	Algoritm nou propus - deep Bi- LSTM S2S pentru vârf de sarcină. Comparăție cu BiLSTM S2S, LSTM S2S, deep LSTM S2S, Levenberg-Marquardt BPNN (LMBP-ANN), și medie Gaussiană SVR(MG-SVR)	Prognoză vârful de sarcină pe ziua următoare. Istoric Mai 2015 - Iulie 2017, 818 - valori pentru antrenament. Fără testare.	Deep Bi-LSTM S2S= 3,67 - 5,92% MG-SVM = 4,74% LMBP-ANN= 4,88% ShallowLSTM: 5,42% Deep LSTM: 8,03% Shallow BiLSTM: 5,15%	[73]

CONTRIBUȚII PERSONALE

3. Importanța temei

Riscurile schimbărilor climatice determină accelerarea tranziției energetice de la combustibili fosili la surse regenerabile, expunând vulnerabilitatea energetică [14]. Aspectele sociale, economice și de mediu necesită o analiză atentă în ceea ce privește dezvoltarea resurselor regenerabile de energie (RES) [15]. Europa are planuri îndrăznețe până în 2050 să devină primul continent neutru din punct de vedere climatic. Direcție formulată de Uniunea Europeană în documentul New Green Deal [1], prin care se dorește ca producția de energie electrică să provină doar din surse regenerabile. Studiile prezentate în [74] și [75] prezintă direcții de tranziție către acest plan ambițios și [76] subliniază contribuția care poate fi adusă de orașele inteligente. Toate aceste planuri durabile ambițioase se confruntă cu limitări ale capacităților de transport [77] și necesită flexibilitate din partea consumatorilor. Conform scenariilor prezentate în [78] și ținând cont de prognozele de consum în Europa, sursele RES se vor confrunta cu întreruperi frecvente din funcționare până în 2050, datorită limitărilor rețelelor electrice.

Industria energetică din Europa este într-o continuă transformare datorită variabilității și schimbărilor climatice, după cum este prezentat în [79]. În aceste condiții rezultă noi provocări și perspective pentru operatorii de sisteme de distribuție, operatorii de sisteme de transport și autoritățile responsabile de reglementări legislative. Sursele de energie electrică regenerabilă au un puternic caracter intermitent, astfel sunt necesare sisteme de stocare a energiei și strategii de control cu o flexibilitate sporită a consumului [80]. Astfel de sisteme pot asigura echilibrarea ofertei și cererii de energie, dar necesită investiții substanțiale. Pentru a oferi o perspectivă mai exactă, autorii prezintă în lucrarea [81] o creștere anuală de până la 25% a cheltuielilor anuale pentru infrastructură energetică pentru a combate efectele schimbărilor climatice în Statele Unite ale Americii.

Evoluția prețului la energie electrică în ultima parte a anului 2020 a demonstrat că factura la electricitate reprezintă un cost ridicat pentru toți utilizatorii finali. Costul ridicat al electricității determină scumpiri în lanț pentru toate produsele și afectează competitivitatea economiei

naționale. Sectorul energetic românesc a devenit pe deplin liberalizat, iar participanții au acces la o varietate de produse energetice. Consumatorii achiziționează energie de la furnizorii care folosesc prognoze pe termen scurt, mediu și lung pentru a tranzacționa pe piața energiei electrice.

În primul semestru al anului 2020, România a importat 3,49 TWh, dublu față de aceeași perioadă a anului precedent, respectiv 1,74 TWh. Producția totală de energie electrică a fost de 24,88 TWh, cu 12% mai mică decât în anul 2019. În același timp, consumul intern a scăzut cu 7%, până la 25,94 TWh. Consumul clienților casnici a crescut cu 2% din cauza restricțiilor pandemice, în timp ce consumul din sectorul industrial a fost mai mic cu 10% [82].

Potrivit Entso-e, în ianuarie 2021, un incident major s-a întâmplat în zona sincronă a Europei Continentale, ceea ce a dus la deconectarea a două regiuni din cauza întreruperii mai multor elemente ale rețelei de transport într-un timp foarte scurt. Mulți consumatori au rămas fără energie electrică și au avut probleme cu nivelurile de tensiune. Acesta este un exemplu relevant care scoate în evidență importanța balanței producție-consum și implementarea conceptului de smart grid [83].

Prognoza consumului de electricitate poate oferi plus valoare provocărilor enumerate mai sus, iar algoritmi cu învățare automată pot revoluționa operarea rețelelor electrice inteligente. Elementul cheie al SG prezentat în [84], [85] și [86] este gestiunea eficientă a unui volum foarte mare de date care să adreseze cerințele de funcționare a rețelelor electrice. Acest obiectiv este un domeniu intens cercetat din cauza impactului financiar și a importanței echilibrării optime a rețelei electrice [87]. Autorii raportului [88] prezintă costurile cu energia electrică ca fiind crucială pentru competitivitatea diverselor industrii pe piețele globale. Importanța prognozei curbelor de sarcină este evidențiată de lucrări precum [89] sau [90], iar în funcție de potențialii beneficiari este prezentat Tabelul 3.1. Prezenta lucrare cercetează integrarea consumatorilor non-rezidențiali în rețele electrice inteligente prin prognoza pe termen scurt și oferă argumente practice pentru angajarea consumatorilor într-o colaborare activă pe piața energiei electrice alături de furnizorul de energie.

Tabel 3.1 Importanța prognozei de consum de electricitate

Activitate	Avantaje	Dezavantaje / Obstacole
Distribuție electricitate	Optimizarea circulației de puteri; Integrarea resurselor regenerabile;	Erori mari și inconsistente prin modele și metode similare (în funcție de perioada de timp, categoria de consum, fereastra de

	Prognoza consumului propriu tehnologic.	prognoză); Resurse suplimentare necesare (timp, angajați calificați, software, instruire).
Transport electricitate	Prognoza consumului național, o planificare mai bună a rețelei de transport.	Erori mari și inconsistente prin modele și metode similare (în funcție de perioada de timp, categoria de consum, fereastra de prognoză); Resurse suplimentare necesare (timp, angajați calificați, software, instruire).
Furnizare de electricitate	Profit maximizat, costuri mai mici pentru echilibrarea portofoliului și angajamentul clientului; comunicare despre planificarea activității.	Erori mari sau inconsecvență în prognozarea sarcinii; Resurse suplimentare necesare (timp, angajați calificați, software, instruire); Furnizorii doresc o utilizare mai mare a energiei.
Consumator	Facturi la electricitate mai mici, eficiență energetică crescută.	Interes scăzut deoarece activitățile principale sunt mai importante decât consumul de energie electrică; Lipsa transparenței; Resurse suplimentare necesare (timp, angajați calificați, software, instruire).

În Figura 3.1 sunt prezentate cantitățile orare ale dezechilibrului național în comparație cu consumul național. Acuratețea prognozei afectează toți participanții la piața energiei electrice. Costurile generate de erorile de prognoză sunt incluse în factura de energie electrică a utilizatorului final printr-un preț pe MWh care compensează expunerea pe piața de echilibrare (PE). În Figura 3.1 dezechilibrul reprezintă diferența dintre valorile prognozate notificate pe piața de echilibrare și valorile măsurate efective puse la dispoziție de operatorul de măsură.

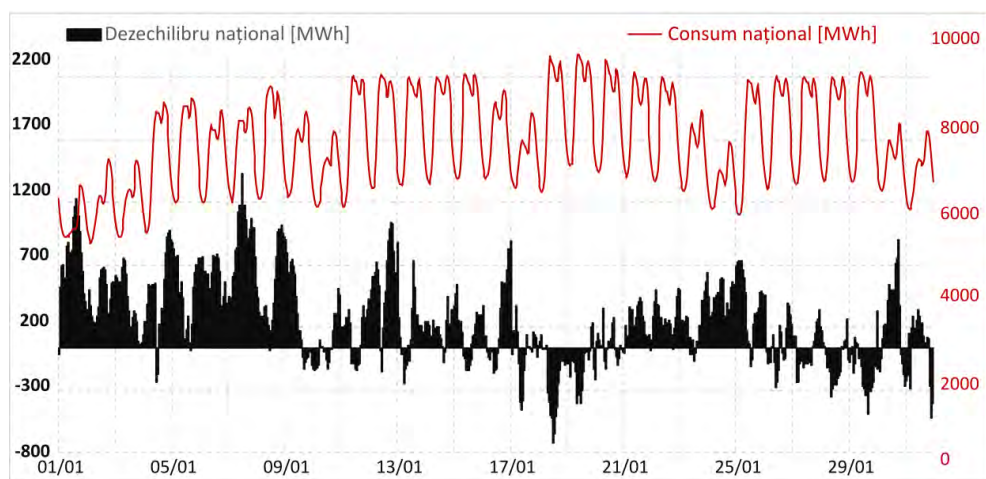


Figura 3.1 Dezechilibrul în SEN în ianuarie 2021

Dezechilibrul total orar de energie electrică din ianuarie 2021 reprezintă o valoare de 4% (219,7 GWh) în consumul național total (5.586,4 GWh) înregistrat în aceeași lună, dar se pot observa pe anumite intervale procente de până la 14,76% (date disponibile pe site-ul operatorului de piață Opcom⁸). Pe termen scurt aceste dezechilibre pot fi minimizezate prin prognoze mai bune. Prin aplicarea algoritmilor cu învățare automată pe baza datelor obținute cu ajutorul tehnologiilor SG, se poate crește potențialul prognozelor. Din acest punct de vedere, putem spune că sinergia dintre SG și prognoza consumului determină o relație reciprocă de optimizare.

În acest studiu, s-au implementat mai multe metode bazate pe învățare automată pentru prognozarea seriilor de timp aferente consumului de electricitate. Studiul compară performanța prognozelor obținute cu douăsprezece metode și se testează diverse combinații pentru variabile independente și corelații autoregresive. Se analizează un grup de consumatori pentru care se simulează achiziția de energie de pe piața liberă. În acest scenariu se reproduce activitatea unui furnizor și se compară rezultatele prognozării curbei de sarcină agregată (întregul grup țintă) cu prognozarea individuală a fiecărui consumator. Istoricul utilizat este aferent anului 2019, iar rezultatele obținute indică în mod consecvent că: (i) algoritmii cu învățare automată sunt potriviți pentru prognoza consumului de energie; (ii) o variație mică în măsura erorilor generează costuri mari pe piața de energie; și (iii) prognozarea curbei de sarcină agregată obține rezultate mai bune decât prognoza individuală. Comparând cea mai bună metodă de prognoză implementată cu abordarea naivă (des utilizată în practică), se obține un cost mai mic cu 9.760 euro pentru trei luni în care au fost testate metodele. Diferența dintre cele mai bune două metode implementate este de 5.326,17 euro. Acest cost trebuie împărțit pentru fiecare consumator din grupul țintă, dar totuși, este un cost care poate fi evitat. Costul generat de dezechilibre este dificil de explicat unui client.

Relația dintre furnizor și consumator poate fi îmbunătățită prin adoptarea tehnologiei SG, în special programe bazate pe controlul curbei de sarcină (demand response) evidențiate de [91] și [92] care datorită beneficiilor potențiale vor determina consumatorii să participe activ la echilibrarea rețelelor electrice. Aceste tehnologii vor îmbunătăți rezultatele prognozelor de consum prin accesul la date în timp real.

⁸ https://www.opcom.ro/tranzactii_rezultate/tranzactii_rezultate.php?lang=ro&id=258

Conform datelor prezentate în [93], sectorul industrial și comercial reprezintă 63,46% din consumul total de energie electrică din lume. Provocarea este de a anticipa comportamentul stocastic al consumatorilor mari și integrarea lor activă în optimizarea circulației de putere din rețeaua electrică. Această lucrare demonstrează că prognozarea curbei de sarcină agregate oferă erori și costuri mai mici pe piața energiei electrice, dar abordarea cu beneficii tehnice este prognozarea fiecărui consumator pe baza proceselor tehnologice specifice. Obstacolul acestei abordări este lipsa contorizării inteligente și a senzorilor dedicați proceselor tehnologice (temperatură, umiditate, etc). Fără date relevante, chiar dacă modelele de prognoză sunt validate, prognozele nu reprezintă o investiție fiabilă pentru sectorul privat. Din punct de vedere financiar este avantajos să se prognozeze curbe de sarcină agregate, dar din perspectiva tehnică a rețelelor electrice, este de preferat să se prognozeze consumatorii individuali pentru a îmbunătăți funcționarea rețelei. Un furnizor poate avea un portofoliu variat de consumatori din toate regiunile țării și astfel prognoza agregată ajută doar la echilibrarea pe piața de echilibrare. Spre deosebire de prognoza agregată, cea individuală poate fi transmisă și către operatorul de sistem în contextul SG.

Piața energiei electrice din România este pe deplin liberalizată. În anul 2009 autorii articolului [94] au prezentat România ca fiind un exportator net de energie electrică în regiune. Zece ani mai târziu, în primele opt luni ale anului 2019, Institutul Național de Statistică [95] a informat că România a importat 2,75 TWh și a exportat 2,48 TWh. În acest context, coroborat cu noile reglementări privind intervalul de decontare la 15 minute de la mijlocul anului 2021, prețurile cresc pentru toate instrumentele pieței de energie.

Prin evidențierea impactului financiar al prognozei există argumente pentru a convinge sectorul industrial și comercial să investească în managementul curbei de sarcină și implementarea prognozei. Cea mai mică eroare de prognoză va genera cel mai mic cost de echilibrare, așa cum se demonstrează în secțiunea cu rezultate, iar o variație mică a erorilor determină costuri ridicate. Analiza se face pe o perioadă fixă cu aceleași prețuri pentru electricitate pentru fiecare metodă de prognoză implementată. Prognoza orară este utilizată pentru a cumpăra energie electrică din PZU. Diferența dintre cantitățile orare notificate pe PZU și consumul real generează surplus sau deficit de energie orar.

4. Ipoteze de lucru - resurse și aplicabilitate

Principala direcție de cercetare din această teză de doctorat se concentrează pe implementarea algoritmilor cu învățare automată pentru prognozarea consumului de energie electrică. Aceștia sunt comparați cu modelele tradiționale de prognoză aplicate consumului de energie. Pentru evaluarea rezultatelor se propune o nouă abordare, complementară indicatorilor statistici pentru calculul erorilor. Algoritmii sunt aplicați pe consumul de electricitate al unui grup de companii private din sectorul industrial și comercial. Metodele de prognoză implementate în această lucrare sunt aplicate pe date orare obținute de la cinci consumatori non-rezidențiali pentru un an întreg (2019). Datele complete de consum aferente fiecărui consumator sunt disponibile doar pentru anul 2019 (conform Tabel 4.1). S-au efectuat mai multe analize și teste pentru a analiza volumul optim de date necesar în procesul de învățare, și s-a stabilit că anul 2019 este suficient și reprezentativ pentru prognoza grupului țintă.

Tabel 4.1 Date reprezentative pentru grupul țintă de consumatori

		Total agregat	Fabrică parchet	Fabrică de mobilă	Super-market	Abator	Dealer Auto
Consum total anual	GWh	37,92	13,78	18,63	3,43	1,70	0,38
Consum orar mediu	MWh	4,33	1,57	2,13	0,39	0,19	0,04
Consum orar minim	MWh	1,56	0,18	0,49	0,16	0,05	0,03
Consum orar maxim	MWh	6,45	2,54	3,32	0,66	0,51	0,10

Pentru implementarea algoritmilor cu învățare automată datele se împart în două seturi: antrenare (80%) și testare (20%). Motivația acestei departajări este antrenarea și validarea algoritmilor. Utilizarea unui număr ridicat de iterații în procesul de antrenare poate să determine o generalizare eficientă a modelului pe datele din setul de antrenare. Setul de testare are rolul de a furniza date care nu au mai fost întâlnite de algoritm pentru a evalua abilitatea acestuia de a prelucra date noi. În procesul de antrenare se folosește metoda ferestrei (intervale) glisante pentru învățarea supravegheată a rețelelor neuronale pentru a memora dependențele pe termen scurt, mediu și lung ale seriilor de timp individuale și agregate. Orizonturile de timp utilizate în lucrare sunt pentru 1 oră, 24 ore, 48 ore și 168 ore.

Utilizarea energiei electrice în industrie permite fabricilor să exploateze resurse pentru a produce bunuri, prin urmare utilizează energia în mod diferit față de entitățile comerciale; i) consumatorul

comercial este o entitate activă în comerț; ii) consumatorul industrial fabrică bunuri, de obicei din materii prime [96]. În figurile următoare sunt prezentate curbele de sarcină ale consumatorilor non-rezidențiali analizați în această lucrare. Curba de sarcină a unei companii producătoare de parchet stratificat este prezentată în Figura 4.1 fiind determinată de caracteristicile echipamentelor electrice și necesităților fluxului tehnologic.

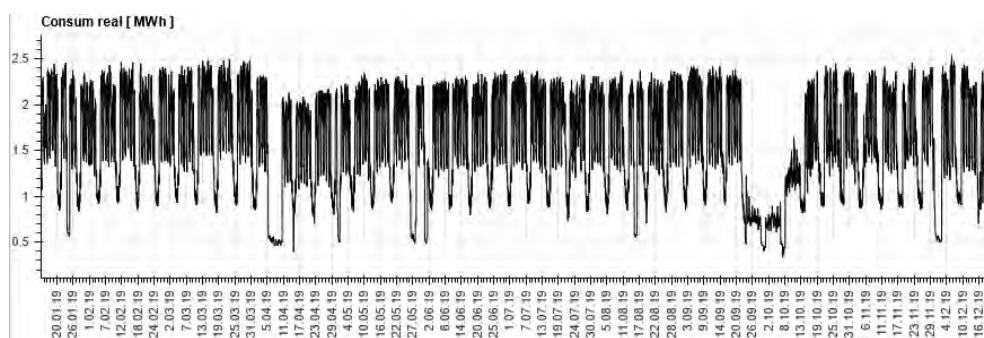


Figura 4.1 Curbă de sarcină pentru fabrica de parchet (FP)

Cel mai mare consumator din grupul analizat, având curba de sarcină anuală (2019) prezentată în Figura 4.2 este o companie activă în industria de prelucrare a lemnului. Programul normal de muncă este în trei schimburi. Alimentarea fabricii este asigurată prin opt transformatoare de putere însumând 11,6 MVA, care alimentează următoarele procese tehnologice, mașini și echipamente:

- instalații care deserveșc echipamentele de tăiere și evacuare;
- instalații care deserveșc sistemul de răcire pentru a asigura frigul necesar păstrării în condiții optime a substanțelor folosite în procesul de spumare;
- instalații care serveșc la prelucrarea și tăierea bureților;
- instalațiile care deserveșc diferitele subsecțiuni la realizarea saltelelor;
- echipamente utilizate pentru realizarea tapițeriei și asamblarea tuturor subansamblurilor;
- instalații de iluminat interior amplasate în perimetrul fizic al tuturor halelor de producție;
- roboți pentru ambalarea produselor finite;
- transportoare pentru transportul produselor în depozitul logistic;
- facilități specifice pentru prepararea alimentelor în cantină;

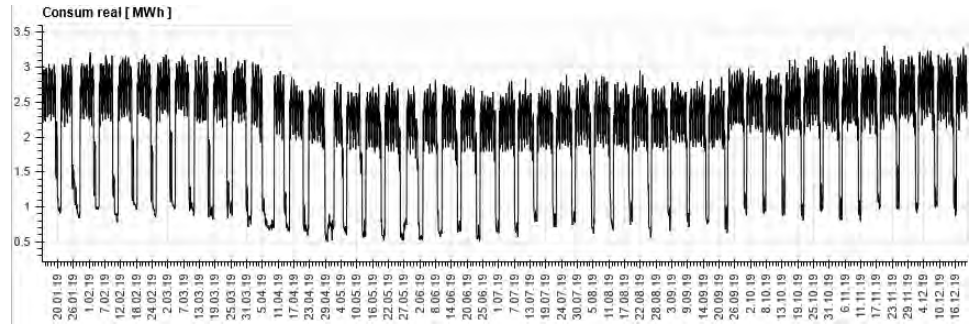


Figura 4.2 Curbă de sarcină pentru fabrica de producție de mobilier (FF)

Din punct de vedere al prognozei, așa cum se poate observa în Figura 4.3 curba de sarcină a unui supermarket este puternic influențată de temperatură în comparație cu consumul unei fabrici de parchet. Programul normal de muncă este în două schimburi.

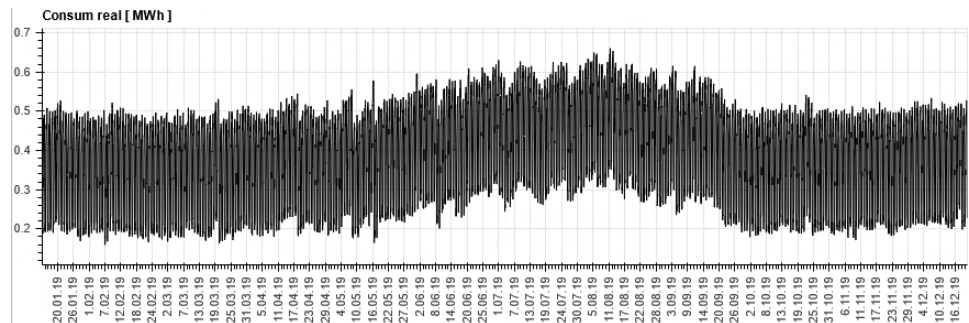


Figura 4.3 Curbă de sarcină supermarket (SM)

În Figura 4.4 este prezentată curba de sarcină a unui consumator reprezentat de o companie din domeniul auto care se ocupă cu comercializare autovehicule și service auto. Energia utilizată este destinată menținerii condițiilor optime pentru clienți în spațiile de prezentare și echipamentele utilizate în atelierul auto, în special în instalațiile de aer comprimat. Programul normal de muncă este de 8 ore.

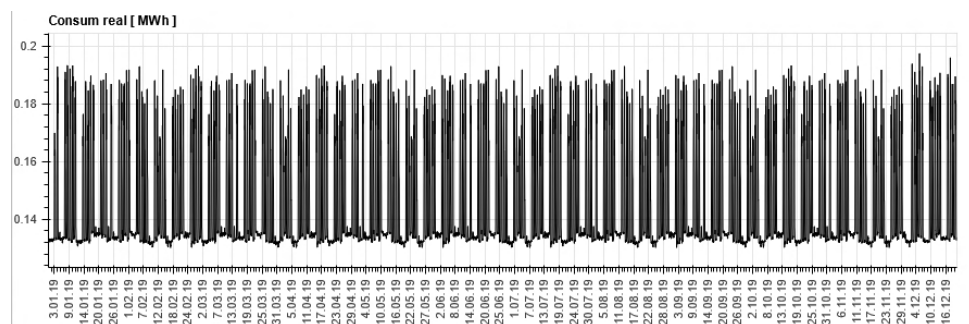


Figura 4.4 Curbă de sarcină pentru auto showroom și service (AS)

Ultimul consumator analizat este o companie producătoare de carne de pui. În Figura 4.5 se prezintă curba de sarcină anuală, programul de muncă al întreprinderii fiind de asemenea un schimb de 8 ore.

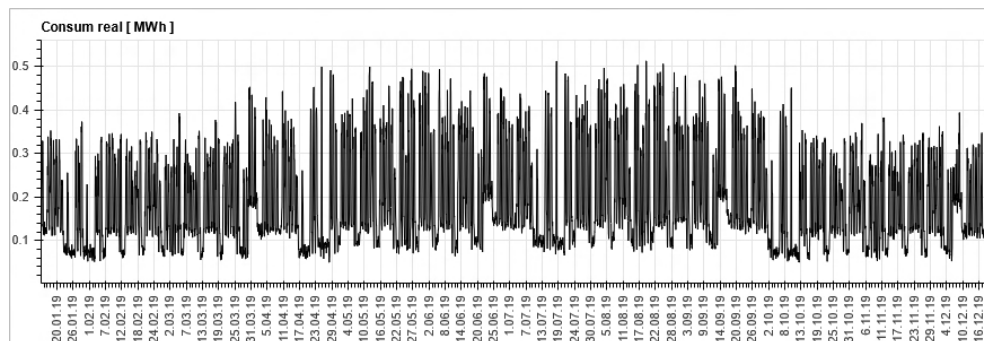


Figura 4.5 Curbă de sarcină abator (FPF)

În Figura 4.6 este prezentată o săptămână normală de funcționare pentru toți consumatorii analizați pentru a evidenția variația orară și zilnică a curbelor de sarcină.

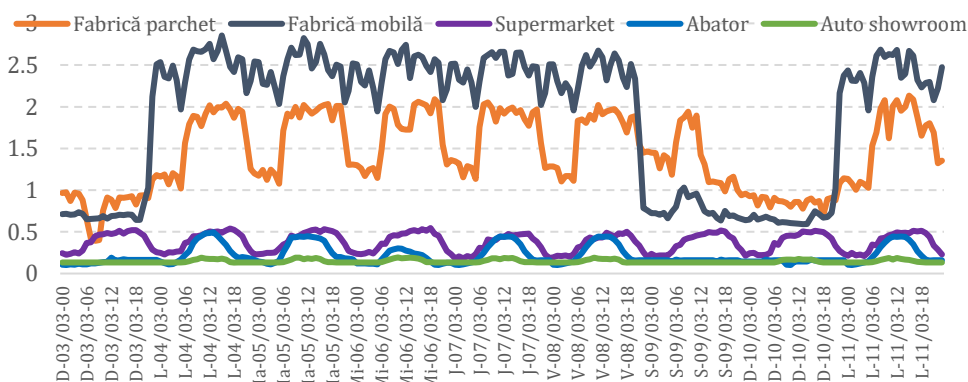


Figura 4.6 Curbă de sarcină – profil săptămânal

Acest cluster de consum însumează la nivel anual 38 GWh și este descris de valorile din Figura 4.7. Clusterul de consumatori conține două entități industriale mari cu procese complexe, un supermarket de aproximativ 2.000 m², o fabrică de procesare a alimentelor și un auto-showroom. Agregând toate curbele de sarcină, se obține o putere orară maximă de 6,5 MW, valoarea medie anuală de 4,5 MW și minimă de 2 MW.

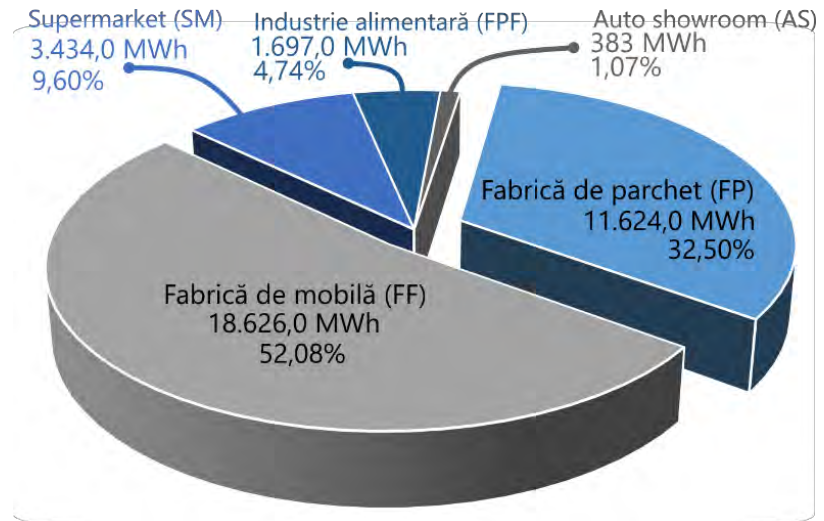


Figura 4.7 Cluster de consumatori analizați.

În această teză se analizează două abordări pentru prognoza curbelor de sarcină: agregată și individuală. În prima abordare se prognozează consumul agregat (se însumează curbele de sarcină înainte de prognoză) conform relației **Error! Reference source not found.**, iar în a doua fiecare consumator este prognozat individual și ulterior se adună prognozele conform relației (4.2). În relații X reprezintă variabilele independente și $t=1..8760$ ore/an. Cele două abordări sunt studiate pentru a evidenția cele mai mici erori de prognoză și abordarea cea mai profitabilă din punctul de vedere al pieței de energie. Se determină că prognozarea individuală a fiecărui consumator are ca rezultat erori mai mari, în comparație cu prognozarea curbei de sarcină agregată.

$$\text{Consum}_{\text{agregat}} = (PF_t + FF_t + SM_t + FPF_t + AS_t)$$

$$\hat{y}_{ag} = f(\text{Consum}_{\text{agregat}}, X) \quad (4.1)$$

$$\hat{y}_{PF_t} = f(PF_t, X_{PF}); \hat{y}_{FF_t} = f(FF_t, X_{FF}); \hat{y}_{SM_t} = f(SM_t, X_{SM});$$

$$\hat{y}_{FPF_t} = f(FPF_t, X_{FPF}); \hat{y}_{AS_t} = f(AS_t, X_{AS})$$

$$\hat{y}_{\text{indiv}} = \hat{y}_{PF_t} + \hat{y}_{FF_t} + \hat{y}_{SM_t} + \hat{y}_{FPF_t} + \hat{y}_{AS_t} \quad (4.2)$$

Se propune metodologia integrată pentru prognoza consumului prezentată în Figura 4.8 pentru reducerea costului de echilibrare pe piața de energie. În figură se prezintă toate etapele premergătoare realizării

prognozei, inclusiv procesul de evaluare. Această metodologie se poate transforma într-o procedură periodică pentru realizarea prognozelor.

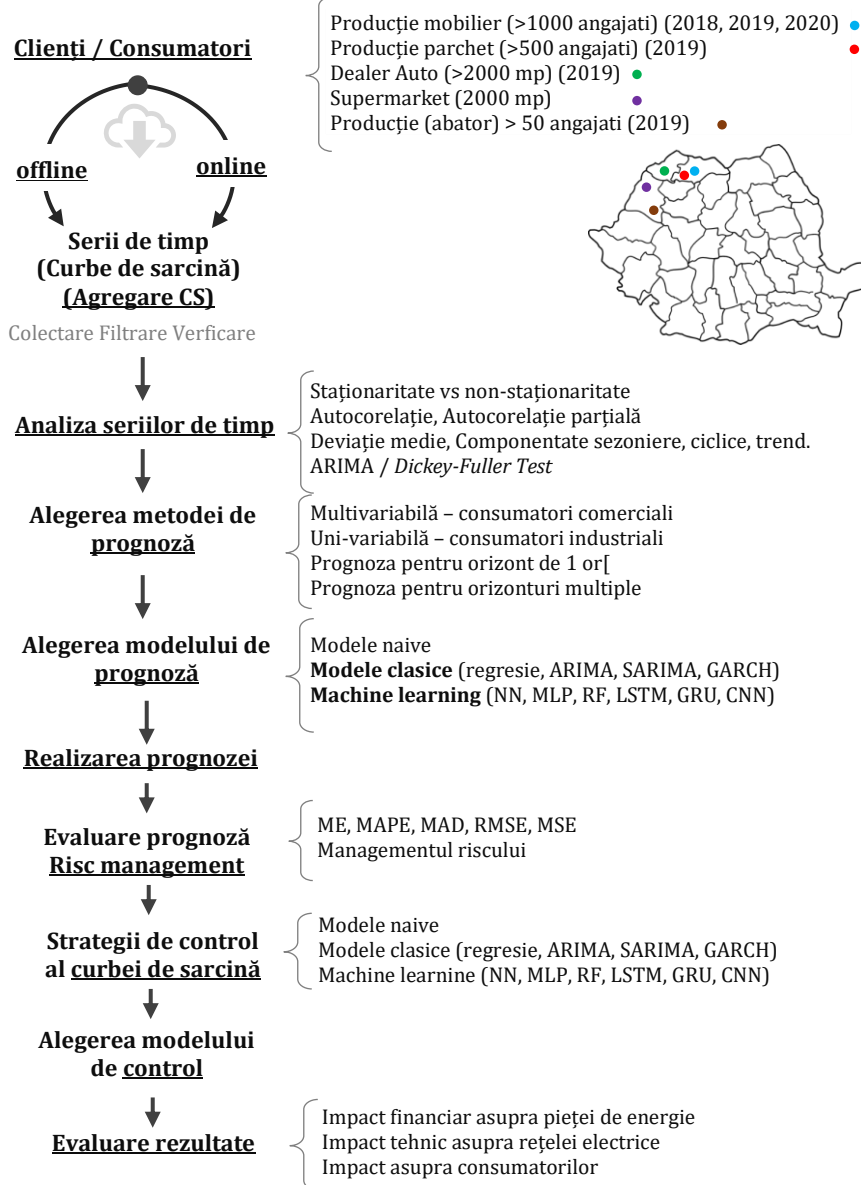
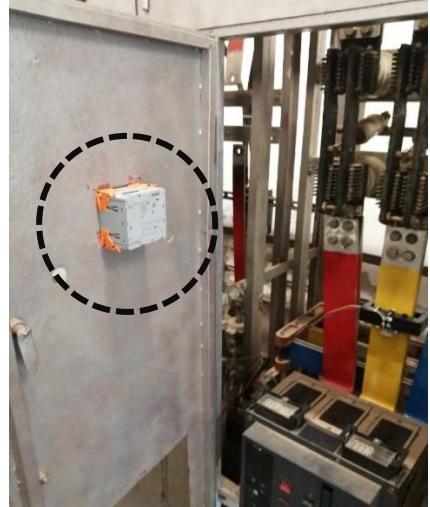

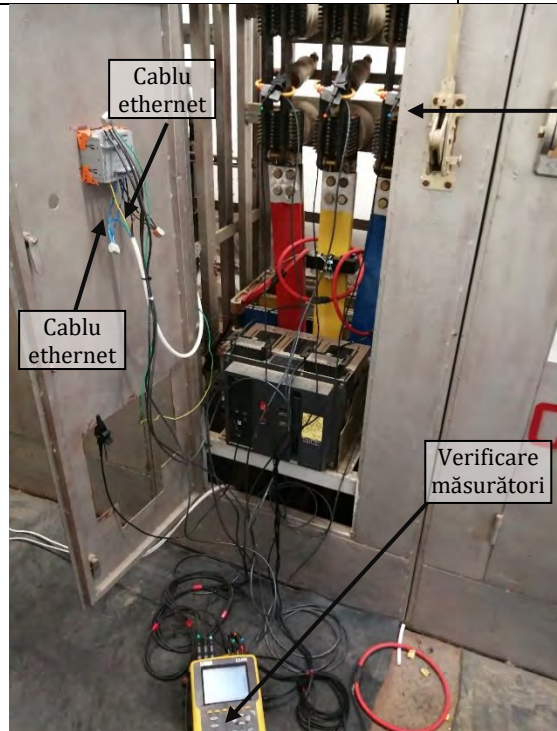



Figura 4.8 Sumarizarea procesului de alegere a metodelor de prognoză

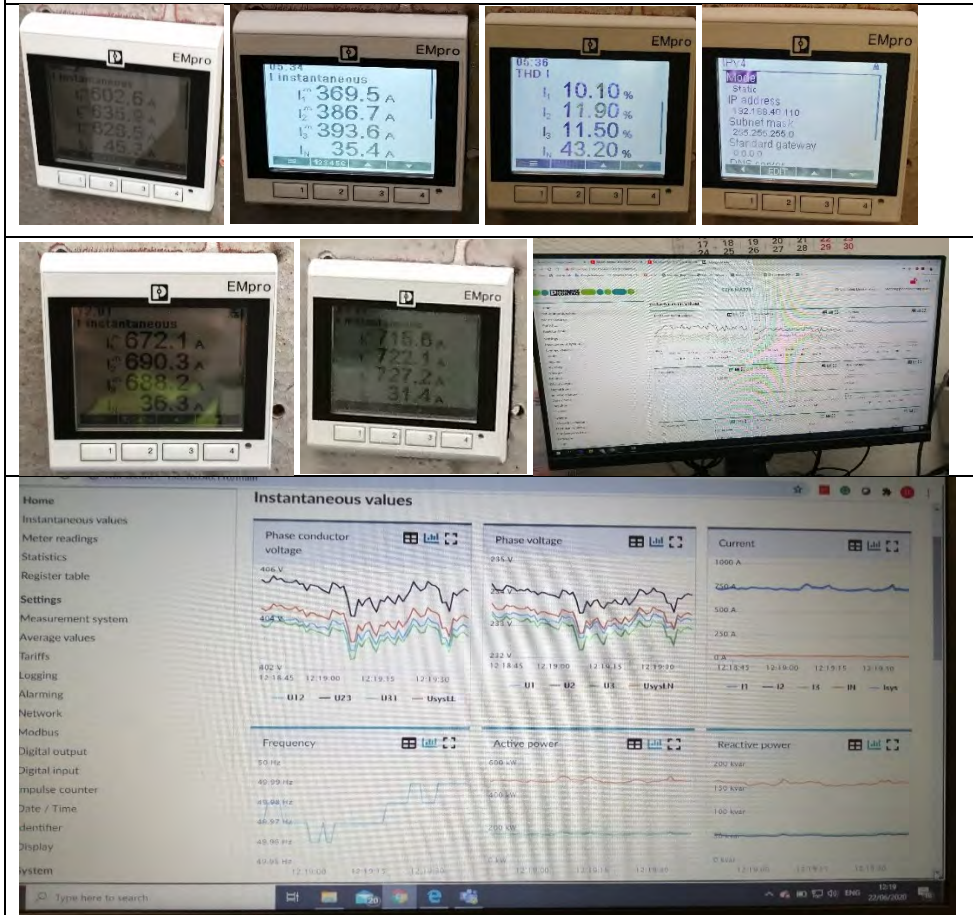
În Tabel 4.2 și Tabel 4.3 se prezintă colaborarea cu compania producătoare de parchet stratificat și modul în care s-au obținut datele necesare prezentului studiu pentru acest consumator.

Tabel 4.2 Instalare centrală de măsură pentru obținerea datelor

Pozitionare centralei de măsură pe ușa tabloului principal	Cablarea și conectarea centralei pentru măsurarea parametrilor electrici
	
	<p data-bbox="1109 840 1364 952">Conectarea cordonelor Rogowski la centrala de măsură (traductori de curent)</p>  <p data-bbox="1061 1288 1252 1545">Conectarea cleștilor de tensiune pentru măsură și alimentarea centralei</p>

Tabel 4.3 Panou de control și interfața pentru accesarea datelor

Capturi cu echipamentul instalat - Capturi cu accesarea datelor de pe orice calculator conectat în rețeaua locală. Pe baza unui webserver se pot accesa toți parametrii electrici măsurați. Datele pot fi accesate în timp real și salvate la interval de 1 min pentru o perioadă de 6 zile.



Scopul inițial al colaborării era să se obțină datele în timp real și transmiterea lor prin internet pentru prelucrare în mediul de programare Python cu algoritmi cu învățare automată, dar din cauza politicilor de confidențialitate și restricțiilor rețelei locale de internet nu s-a reușit în timp util să configurăm conexiunea pentru a satisface cerințele de securitate ale companiei.

Motivul pentru care s-a folosit acest set de date se datorează opțiunilor multiple disponibile pentru controlul consumului și simularea situațiilor reale care se întâmplă în practică. Seriile de timp analizate au o variație ridicată în jurul valorilor medii și fluctuații iregulare ale seriilor de timp. Cu cât se agregă (însurează) mai mulți consumatori se obțin

serii de timp mai ușor prognozabile, astfel s-a lucrat pe cel mai defavorabil scenariu. În Figura 4.9 se evidențiază conturul energetic și bilanțul electro-energetic al consumatorului PF. Scopul diagramelor este să prezinte complexitatea consumatorilor și analiza realizată pentru determinarea variabilelor folosite pentru prognoză. Figura 4.9 scoate în evidență și potențialul de control al consumatorului, iar valorile din diagrama Sankey (Figura 4.10) sunt determinate prin măsurători cu analizorul de calitate Chauvin Arnoux Qualistar+ C.A 8336.

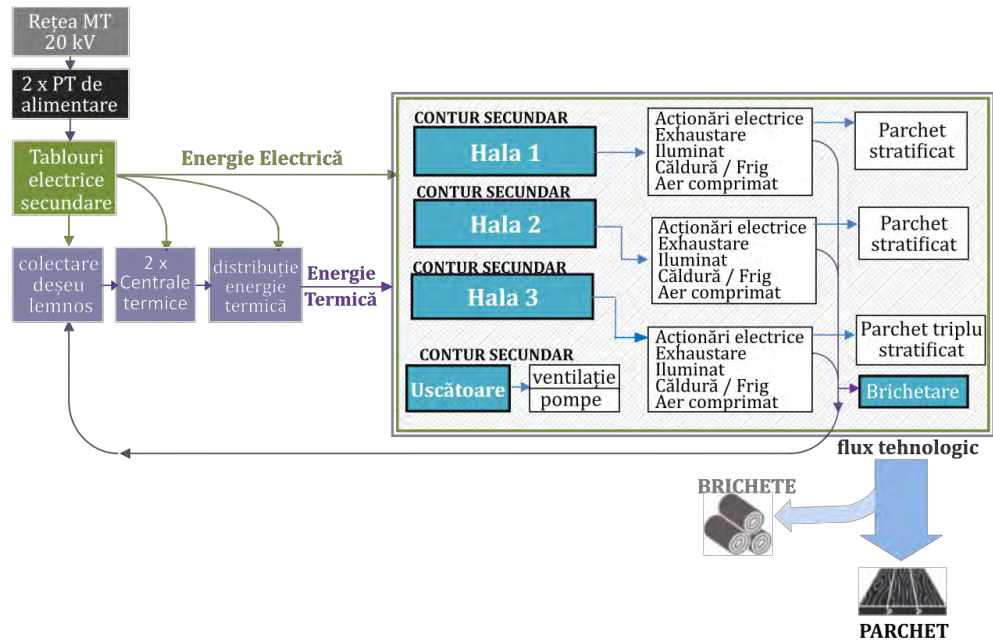


Figura 4.9 Consumatorul industrial – fabrică de parchet

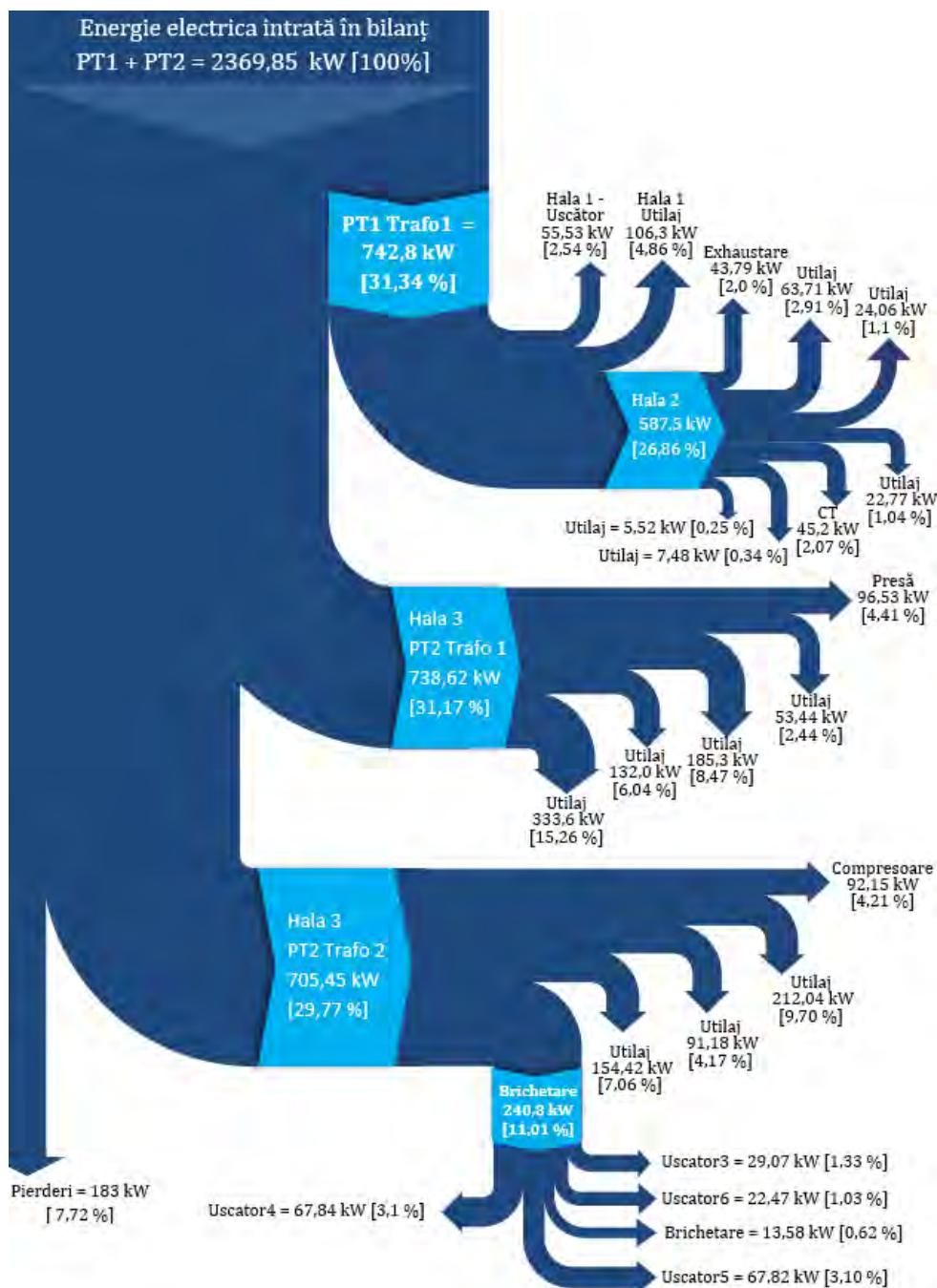


Figura 4.10 Diagrama Sankey – fabrica de parchet (FP)

În Figura 4.11 se prezintă conturul energetic al celui mai mare consumator din grupul țintă și repartizarea consumului pe fiecare post

de transformare. Această companie nu are instalat sistem de monitorizare, datele utilizate în această lucrare fiind obținute de la furnizor și prin măsurători cu analizor de calitate. În Figura 4.12 se prezintă bilanțul electroenergetic pentru supermarket-ul analizat în această lucrare.



Figura 4.11 Consumatorul industrial – fabrică de mobilă

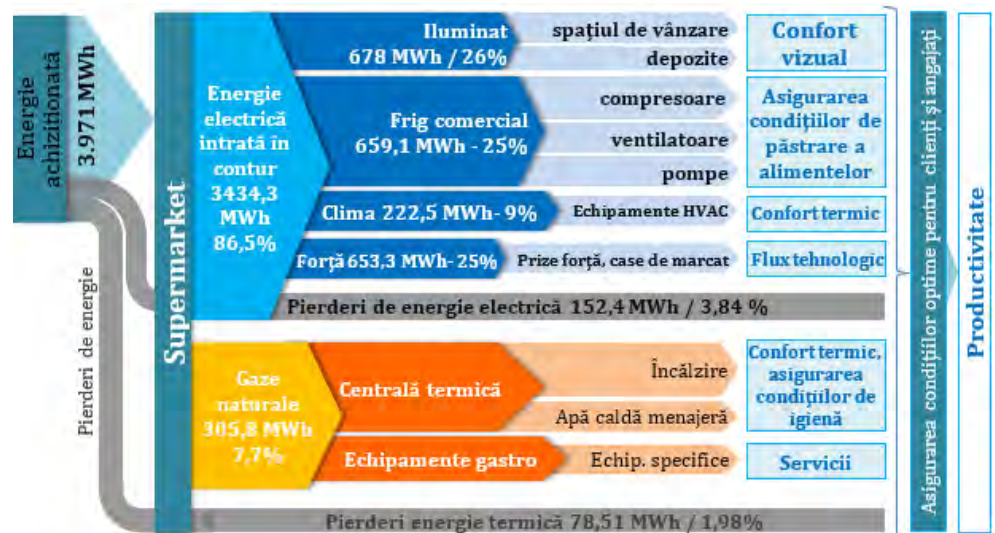


Figura 4.12 Consumatorul comercial – supermarket (bilanț real anual)

În Figura 4.13 se evidențiază evoluția curbei de sarcină pentru consumul agregat orar, atât evoluția anuală cât și pentru o perioadă mai scurtă de trei luni, pentru ilustrarea formei curbei de sarcină orară.

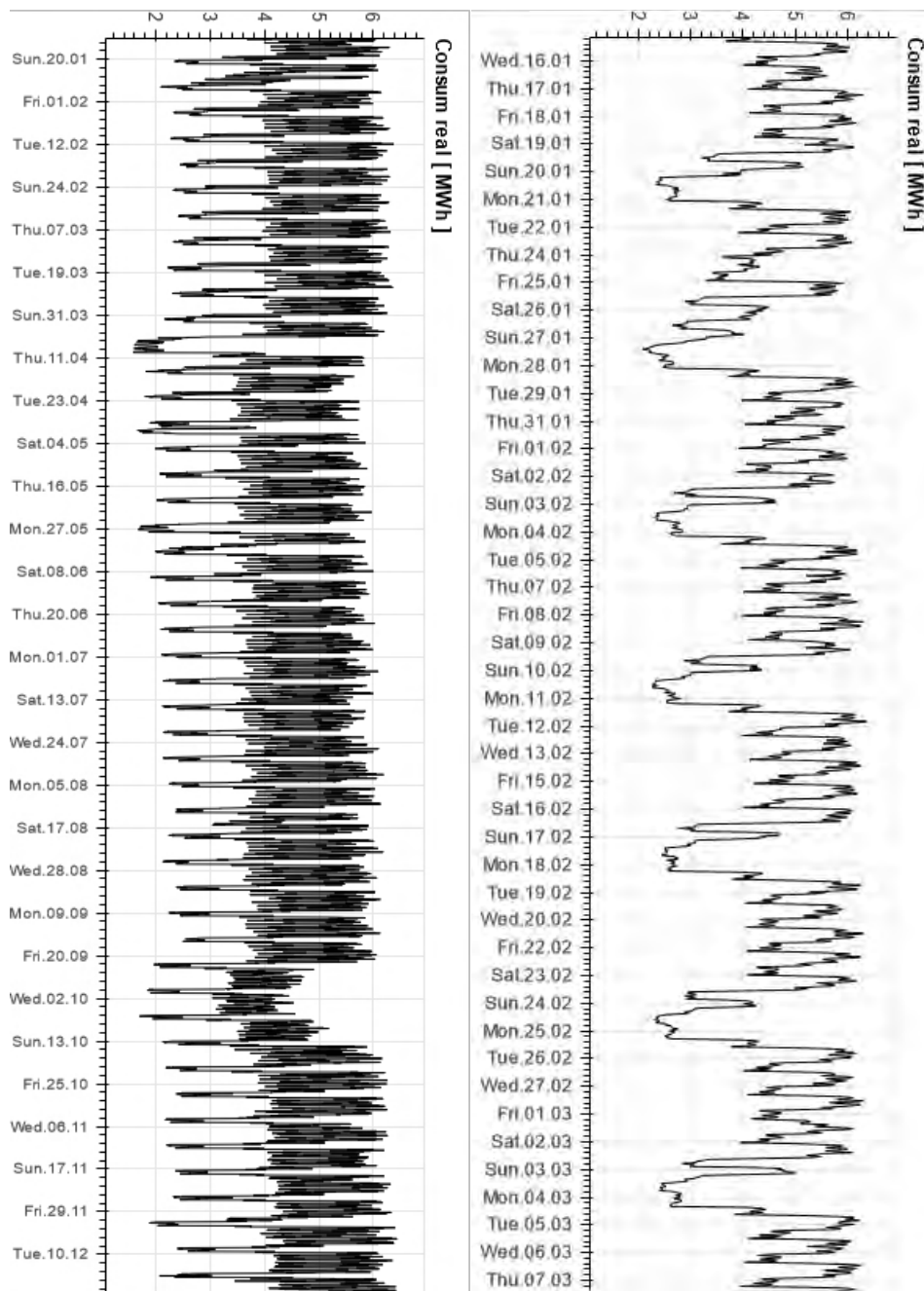


Figura 4.13 Curbele de sarcină agregate ale consumatorilor

Repartizarea consumului orar pe secții și utilaje sau echipamente importante ajută la identificarea comportamentului energetic. Scopul diagramei Sankey este de a identifica punctele de consum care necesită monitorizare. Pentru integrarea consumatorului în smart grid este necesar un nivel ridicat de monitorizare pentru a putea realiza prognoze eficiente și pentru implementarea soluțiilor de demand response.

Comportamentul energetic al fiecărui consumator depinde de totalitatea echipamentelor necesare procesului de producție. S-a analizat fiecare locație în parte pentru a determina factorii care influențează curbele de sarcină. Spre exemplu, în cazul fabricilor care activează în sectorul industriei prelucrătoare de lemn, s-a identificat umiditatea ca fiind un factor important care influențează consumul deoarece în ambele locații se usucă lemn prin intermediul unor uscătoare tip depozit. Consecința unei umidități ridicate în aer este un consum mai ridicat de energie pentru uscarea lemnului față de zilele cu umiditate normală.

În articolul [97] se prezintă metode de clasificare și analize similare cu cele efectuate în această teză pentru consumatori, dar cu scopul de a crea profile de sarcină electrică pentru diferite comportamente de consum. În prezentul studiu, se lucrează cu trei fabrici producătoare de bunuri și doi consumatori comerciali de tipuri diferite. Din discuțiile cu consumatorii, s-au identificat probleme similare, așa cum este subliniat în lucrarea [98] consumatorii industriali și comerciali au puțină conștientizare despre amprenta energetică; în plus, s-a identificat un interes scăzut pentru costurile cu energia, care este considerat un cost necontrolabil. Autorii din [99] prezintă comportamentul consumatorilor industriali și comerciali și elasticitatea prețurilor. Consumatori clasificați în diverse categorii - interne; mic comerț; comerț mediu; comerț mare; comerț mediu și mare industrial [100]. Autorii din [101] au analizat douăzeci de curbe de sarcină cu un nivel de tensiune de 10 kV și au arătat că rezultatele clasificării curbelor de sarcină pot fi utilizate ca referință pentru sprijinirea deciziilor în elaborarea politicilor privind prețul energiei electrice. În Figura 4.14 se prezintă caracteristicile utilizate în prognoză și corelația în timp cu curbele de sarcină agregate.

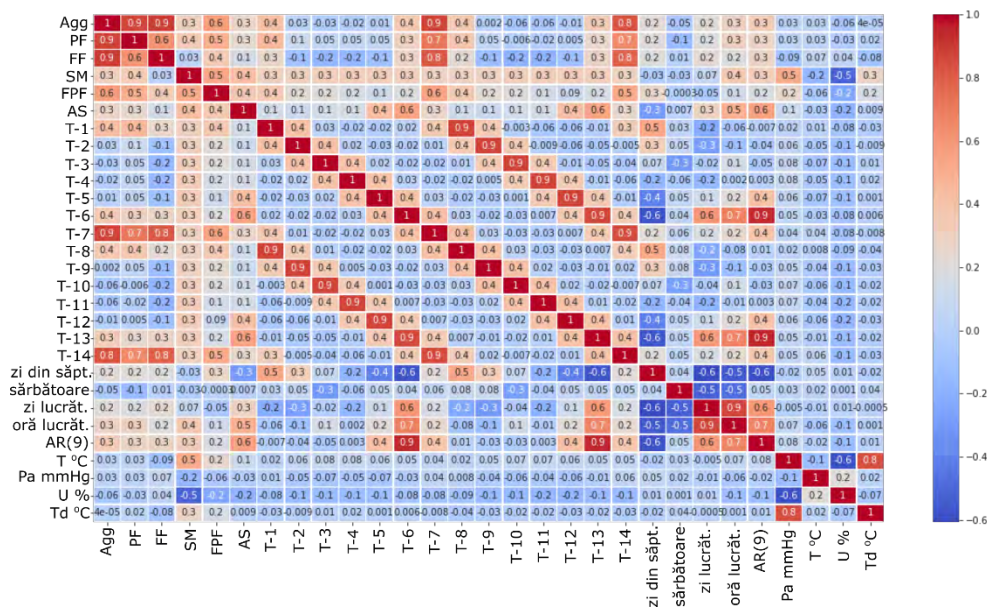


Figura 4.14 Variabile exogene și autoregresive

Având în vedere poziția geografică a consumatorilor analizați, localizați în regiunea nord-vest, s-au considerat în prognoză date orare meteorologice de la stația meteorologică instalată la aeroportul internațional Maramureș din Baia-Mare. Variabilele exogene luate în considerare în analize și prognoză sunt prezentate sintetic în Tabel 4.4 și Tabel 4.5. În prognoză se utilizează doar variabilele exogene care influențează direct consumul conform analizelor statistice sau care îmbunătățesc rezultatele prognozelor.

Tabel 4.4 Variabile exogene meteorologice luate în calcul pentru prognoză

T	Temperatura la doi metri de la sol	°C
Po	Presiune atmosferică la nivelul stației meteo (2 m)	mmHg
P	Presiune atmosferică la nivelul mării (2 m)	mmHg
Pa	Schimbarea presiunii atmosferice la nivelul ultimilor 3 ore (2 m)	unități
U	Umiditate relativă la doi metri de la sol	%
DD	Direcția vântului	unități
Ff	Viteza vântului	m/s
N	Acoperirea cerului	%
Td	Punctul de rouă	°C

Tabel 4.5 Valorile meteorologice orare folosite pentru prognoză⁹

	T	Po	P	Pa	U	DD	Ff	N	Td
01/01/19 0:00	0,8	733,2	771,7	0,8	100	(2) Vest	(1m/s)	100%.	0,8
01/01/19 1:00	0,7	733,3	771,9	0,7	100	(1,5) Nord-	(1m/s)	100%.	0,7
01/01/19 2:00	0,6	733,5	772	0,5	100	(1,5) Nord-Vest	(1m/s)	100%.	0,6
01/01/19 3:00	0,5	733,9	772,6	0,7	100	(2) Vest-Sud	(1m/s)	100%.	0,5
01/01/19 4:00	0,3	733,8	772,5	0,5	100	(2,5) Vest-Sud	(1m/s)	90 - 100%	0,3
01/01/19 5:00	0	733,6	772,3	0,1	100	(2) Vest	(1m/s)	90 - 100%	0
01/01/19 6:00	0,2	733,3	772	-0,6	98	(1,5) Nord-	(1m/s)	90 - 100%	-0,1
01/01/19 7:00	0,3	733,3	771,9	-0,5	96	(1) Nord	(1m/s)	90 - 100%	-0,2
01/01/19 8:00	0,1	733,3	772	-0,3	96	(1,5) Nord-	(1m/s)	70 - 80%.	-0,5
01/01/19 9:00	-0,2	733,3	772	0	99	(2,5) Sud-Vest	(1m/s)	100%.	-0,4
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
31/12/19 11:00	0,9	732,5	771,1	-0,8	90	(3) Sud	(1m/s)	70 - 80%.	-0,6
31/12/19 12:00	2,6	732,1	770,4	-1,2	81	(1,5) Nord-	(1m/s)	90 - 100%	-0,4
31/12/19 13:00	2,4	731,5	769,8	-1,5	80	(1,5) Nord-	(1m/s)	50%.	-0,8
31/12/19 14:00	2,5	730,8	769	-1,7	77	(1,5) Nord-Vest	(1m/s)	70 - 80%.	-1,2
31/12/19 15:00	2,3	730,3	768,5	-1,8	84	(4) Est	(2m/s)	90 - 100%	-0,2
31/12/19 16:00	2,2	729,7	767,9	-1,8	81	(3,5) Sud-Est	(1m/s)	90 - 100%	-0,8
31/12/19 17:00	1,7	729,2	767,4	-1,6	85	(3,5) Sud-Est	(1m/s)	90 - 100%	-0,6
31/12/19 18:00	1,2	728,6	766,9	-1,7	89	(3,5) Sud-Est	(1m/s)	50%.	-0,5
31/12/19 19:00	-0,1	728,1	766,6	-1,6	96	0	(1m/s)	50%.	-0,7
31/12/19 20:00	-0,3	727,6	766	-1,6	97	(1) Nord	(1m/s)	70 - 80%.	-0,7

Toți acești factori sunt analizați deoarece influențează consumul de energie electrică. Temperatura are un impact considerabil în consum, în special pentru consumatorii comerciali. Umiditatea este relevantă pentru consumatorii industriali, deoarece prelucrarea lemnului se face la o umiditate optimă care este obținută cu ajutorul unor uscătoare speciale, dedicate pentru cantități mari de lemn. Corelația cu temperatura este valabilă numai pentru consumatorul supermarket, pentru restul consumatorilor nu au fost identificate corelații semnificative cu temperatura. Fabrica de prelucrare a alimentelor are un depozit frigorific de 1.000 m², dar nu există corelație între consumul total al unității cu temperatura exterioară. Lipsa corelației cu temperatura există din diverse motive tehnice și organizatorice, care reflectă un comportament de consum variabil strict dependent de necesitățile de producție. Lucrarea prezentată de [104] explică modul de analiză a consumului total de energie electrică a mai multor consumatori și extragerea componentelor cheie precum încălzirea, ventilația și aerul condiționat (HVAC), iluminatul rezidențial și consumul de iluminat stradal din consumul total. Pentru gestionarea energiei în clădiri de birouri, autorii articolului [102] propun o abordare interesantă cu

⁹ [https://rp5.ru/Arhiva_meteo_%C3%AEn_Baia_Mare_\(aeroport\)](https://rp5.ru/Arhiva_meteo_%C3%AEn_Baia_Mare_(aeroport))

lanțuri Markov pentru prognoza de consum și producție fotovoltaică (12,24 kWp) cu o eroare absolută medie măsurată de 34W. Comportamentul energetic al magazinelor mari (supermarket și hypermarket) este analizat în [103] și se prezintă o prognoză până în 2040 pentru acest sector: consumul de electricitate va crește cu până la 5,5%, iar consumul de gaze naturale se estimează că va scădea cu 28%.

Prognozele agregate și individuale realizate în teză sunt folosite pentru a simula achiziția de energie electrică de pe piața pentru ziua următoare (PZU), iar erorile de prognoză sunt corelate cu cantitatea și costul pe piața de echilibrare. În acest mod, se evaluează impactul orar al prognozelor asupra pieței energiei electrice.

Resurse software & hardware

Implementarea metodelor de prognozare se face în mediul de dezvoltare Python. Algoritmii cu învățare automată (ML) utilizați pentru prognoză și prezentați în capitolul următor, respectiv diagrama pseudo-cod rezumă pe scurt arhitectura sistemului implementat ierarhic, pentru metodele de învățare automată dezvoltate. Pentru implementarea algoritmilor, s-a folosit Tensorflow [105], care este o bibliotecă open-source dezvoltată pentru aplicații tradiționale și de învățare profundă. Keras [106] este o bibliotecă open-source API de nivel înalt pentru învățarea automată care funcționează deasupra Tensorflow. În Figura 4.15 se evidențiază o comparație care a fost la baza alegerii instrumentelor folosite în această teză.

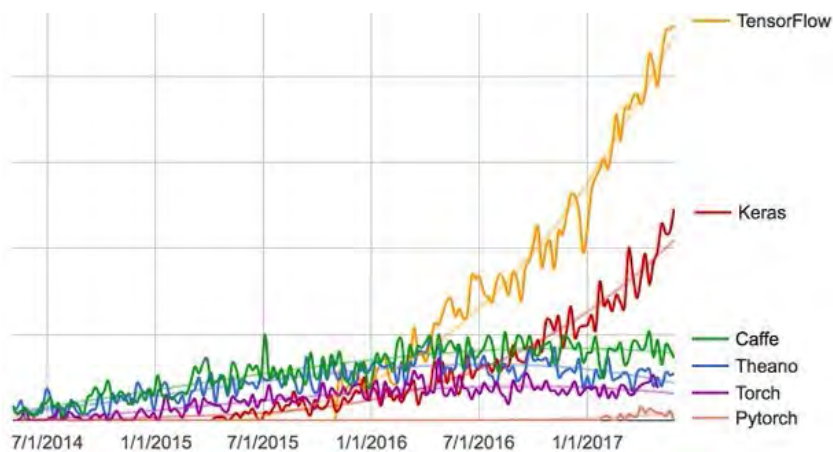


Figura 4.15 Interesul de căutare pentru platforme ML¹⁰

¹⁰ <https://twitter.com/fchollet/status/871089784898310144?lang=cs>

Pentru pregătirea datelor și vizualizarea ulterioară a rezultatelor, au fost utilizate bibliotecile Scikit-learn [107], Numpy [108] și Matplotlib [109], precum și bibliotecile Seaborn [110]. Simulările sunt efectuate pe un calculator Intel (R) Core (TM) i5-4690K CPU@3.5GHz, RAM 16 GB, sistem de operare pe 64 de biți, procesor bazat pe x64. În Tabel 4.6 se prezintă o comparație între cele mai utilizate platformele pentru inteligență artificială.

Tabel 4.6 Platforme pentru învățare aprofundată¹¹

Software	Date	Active	Open Source	Cod	CUDA	Licență	Creator
Wolfram Mathematica	1988	Yes	No	C++, Wolfram Language, CUDA	Yes	Owner	Wolfram Research
MATLAB+Deep Learning TB	1992	Yes	No	C, C++, Java, MATLAB	Yes	Owner	MathWorks
Dlib	2002	N/A	Yes	C++	Yes	Boost Software	Davis King
Torch	2002	No	Yes	C, Lua	Yes	BSD	Ronan Collobert, Koray Kavukcuoglu, Clement Farabet
OpenNN	2003	N/A	Yes	C++	Yes	GNU LGPL	Artelnics
Intel Math Lib.	2003	Yes	No		No	Owner	Intel
Theano	2007	No	Yes	Python	Yes	BSD	University of Montreal
Neural Designer	2012	N/A	No	C++	No	Owner	Artelnics
Caffe	2013	No	Yes	C++	Yes	BSD	Berkeley Vision and Learning Center
Deeplearning4j	2014	N/A	Yes	C++, Java	Yes	Apache 2.0	SkyMind engineering team; Deeplearning4j community; originally Adam Gibson
Intel DAAL	2015	N/A	Yes	C++, Python, JS	No	Apache 2.0	Intel
Apache SINGA	2015	N/A	Yes	C++	Yes	Apache 2.0	Apache Software Foundation
Chainer	2015	No	Yes	Python	Yes	BSD	Preferred Networks
Keras	2015	Yes	Yes	Python	Yes	MIT	François Chollet
Apache MXNet	2015	Yes	Yes	C++ core lib	Yes	Apache 2.0	Apache Software Foundation
TensorFlow	2015	Yes	Yes	C++, Python, CUDA	Yes	Apache 2.0	Google Brain
BigDL	2016	N/A	Yes	Scala	No	Apache 2.0	Jason Dai (Intel)
Microsoft Cognitiv(CNTK)	2016	No	Yes	C++	Yes	MIT	Microsoft Research
PyTorch	2016	Yes	Yes	Python, C, C++, CUDA	Yes	BSD	Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan (FB)
Flux	2017	Yes	Yes	Julia	Yes	MIT	Mike Innes
PlaidML	2017	Yes	Yes	Python, C++, OpenCL	No	Apache 2.0	Vertex.AI, Intel

¹¹<https://towardsdatascience.com/top-5-deep-learning-frameworks-to-watch-in-2021-and-why-tensorflow-98d8d6667351>

5. Metodologia generală pentru prognoză

Prezenta lucrare se focalizează pe utilizarea algoritmilor cu învățare automată pentru prognozarea consumului de energie electrică în strânsă comparație cu modelele statistice tradiționale și propune o nouă abordare pentru evaluarea prognozei. Metodologia generală este prezentă sintetic în Figura 5.1. Se propune o abordare nouă care constă în modelarea prețurilor și structurii pieței energiei electrice prin transformarea erorilor de prognoză în costuri financiare (achiziție și echilibrare). Prin urmare, abordarea determină dacă valorile orare prezise sunt fiabile pentru achiziția de energie. Erorile mari de prognoză pot genera costuri de echilibrare ridicate, făcând acest efort redundant pentru furnizor. Lucrarea se concentrează mai mult pe rezultate și încearcă să găsească cea mai bună soluție la problemele de prognoză în ceea ce privește viteza, acuratețea și simplitatea.

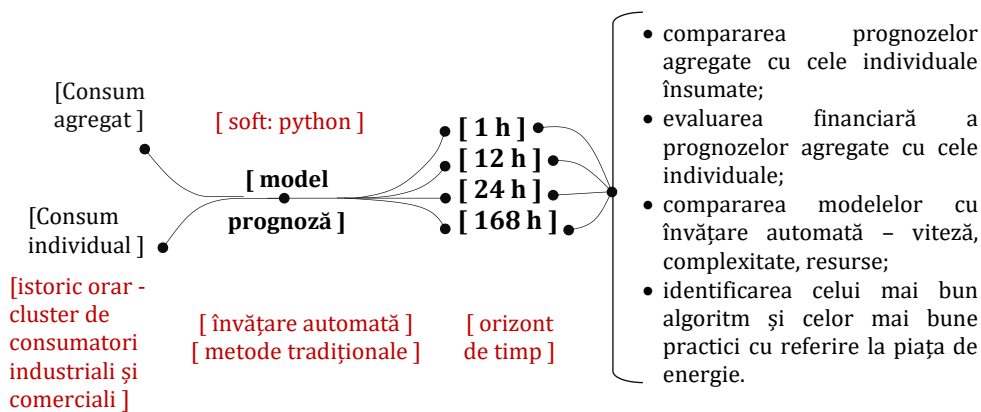


Figura 5.1 Metodologia generală a lucrării

5.1. Prognoză bazată pe metode tradiționale

Un punct de referință pentru analiza seriilor de timp este testul Dickey – Fuller [111] care reprezintă ipoteza nulă pentru metoda de regresie și indică faptul că seriile de timp analizate prezintă staționaritate. În Tabel 5.1 prezentăm testul de non-staționaritate pentru seriile de timp care reprezintă curbele de sarcină pentru fiecare consumator analizat.

Tabel 5.1 Analiză statistică Dickey–Fuller.

Indicator	Agg	FP	FF	SM	FP	AS	
ADF Statistic:	-11,995	-11,421	-11,70863	-14.227	-10,572067	-13,953997	
p-value:	0,00	0,0000000	0,006772	0,00	-10,572067	-13,953997	
Valori critice:	1%	-3,431	-3,431	-3,431	-3,431	-3,431	-
	5%	-2,862	-2,862	-2,862	-2,862	-2,862	-2,862
	10%	-2,567	-2,567	-2,567	-2,567	-2,567	-2,567
Respingere ipoteză - TS	Staționar	Staționar	Staționar	Staționar	Staționar	Staționar	

Agg = consum agregat; FP = fabrică de parchet; FF = fabrică de mobilă;
SM =supermarket; FPF = fabrică procesare alimente; AS = auto dealer showroom

Caracteristica de autocorelație a seriilor de timp aferente curbelor de sarcină este evidențiată cu ajutorul graficelor de distribuție a datelor pentru factorul unitar de întârziere. Pe măsură ce nivelul de autocorelație crește, punctele se adună mai strâns de-a lungul liniei de regresie. În Figura 5.2 se prezintă analiza consumului de energie și putem observa că pentru AS există un nivel ridicat de neliniaritate. Rezultatele prognozelor sunt mult mai bune atunci când sunt aplicate pe seriile cronologice cu un tipar repetitiv (cazul supermarket-ului), din cauza autocorelării din datele istorice anterioare.

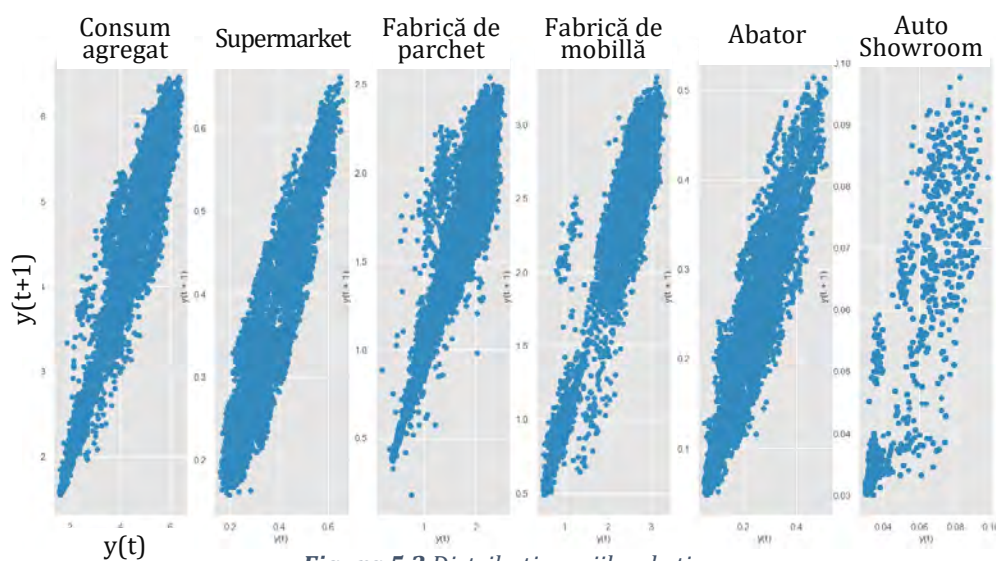


Figura 5.2 Distribuția seriilor de timp

Pentru a avea o referință de comparație pentru prognoza curbei de sarcină pe termen scurt (PTS), s-au implementat modele simple

pentru a avea un punct de plecare în compararea rezultatelor și s-a arătat necesitatea unor abordări mai complexe. S-au implementat metode naive, prezentate în capitolul 2.3.1:

- i) Valori orare reale din ziua anterioară sunt folosite ca prognoză pe ziua următoare, ecuația (5.1),

$$\hat{y}_h = y_{h-24} \quad (5.1)$$

unde, $h = 1 \dots 8760$. Pentru sfârșitul de săptămână au fost utilizate valorile orare anterioare pentru sâmbătă și duminică. Pentru prognoza de luni, s-au luat în considerare valorile de vinerea anterioară. Pentru prognoza sărbătorilor, a fost construit un profil mediu. După o perioadă lungă de vacanță, ultimele zile lucrătoare au reprezentat următoarele prognoze.

- ii) Prognoza orară pentru ziua următoare este egală cu valorile orare reale din aceeași zi a săptămânii trecute, ecuația (5.2).

$$\hat{y}_h = y_{h-168} \quad (5.2)$$

unde, $h > 168$ ore. Valorile orare din săptămâna precedentă devin prognozate pentru săptămâna curentă în mod corespunzător. De exemplu, pentru prognozarea valorilor orare viitoare pentru ziua de marți, folosim valorile orare aferente zilei de marți din săptămâna anterioară.

Media mobilă aplicată pentru valorile orare din ziua anterioară și în urmă cu două săptămâni, sunt conform ecuației (5.3):

$$\hat{y}_h = \frac{y_{h-24} + y_{h-168} + y_{h-336}}{3} \quad (5.3)$$

În modelul autoregresiv, consumul de energie electrică din zilele anterioare este utilizat pe post de predictor. O relație autoregresivă exprimă corelația dintre un număr de valori precedente din serie care prognozează valoarea în prezent. Modelul autoregresiv utilizat este AR (ordin 14), ceea ce înseamnă că se ține cont de ultimele două săptămâni de date pentru construirea ecuației de regresie (5.4).

$$\hat{y}_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \dots + \beta_{14} y_{t-14} \quad (5.4)$$

Analiza de autoregresie s-a realizat atât în Excel [112] cât și în Python folosind librăriile Statsmodels [113]. Analiza realizată pentru curbele de sarcină cunoscute, descrie relația dintre variabile independente (consum trecut $t-1$) și variabila dependentă (consumul în ziua t). Astfel se obține o ecuație în care coeficienții reprezintă relația dintre consumul trecut și cel prezent. În urma acestei analize se obțin o serie de indicatori statistici prezentați în Tabel 5.2. Valoarea R^2 (coeficientul lui Pearson ridicat la pătrat) este egal cu 0,778, ceea ce evidențiază că 77,8% din variația consumului prezent este corelată cu consumul trecut. Pe baza punctajului valorii probabilitice P -value, pentru construirea ecuației de autoregresie folosită în prognoză se păstrează doar valorile orare zilnice relevante pentru ecuația regresiei (5.4). Astfel factorii autoregresivi $T - h$ care au obținut P -value $> 0,05$ nu sunt folosiți în ecuația de regresie folosită la prognoză, deoarece nu există corelație cu consumul. În Tabel 5.2 se poate observa o corelație puternică între consumul din ziua T cu consumul din zilele $T-1$, $T-6$, $T-7$, $T-8$, $T-14$, ceea ce indică un tipar de consum similar pentru aceleași zile din săptămână. Spre exemplu comportamentul energetic din ziua de luni este diferit de cel de sâmbătă și nu există o relație liniară între variația seriilor de timp.

Tabel 5.2 Analiză statistică. $R^2 = 0,778$

		Coeficient	Eroare standard	t Stat	p-value	inferior	superior	inferior	superior
						95%	95%	95.00%	95.00%
	β_0	0,334	0,049	6,768	0,000	0,237	0,431	0,237	0,431
T-1	β_1	0,370	0,011	32,485	0,000	0,348	0,392	0,348	0,392
T-2	β_2	-0,016	0,012	-1,277	0,202	-0,040	0,008	-0,040	0,008
T-3	β_3	0,039	0,012	3,220	0,001	0,015	0,063	0,015	0,063
T-4	β_4	0,071	0,012	5,791	0,000	0,047	0,095	0,047	0,095
T-5	β_5	-0,008	0,012	-0,653	0,514	-0,032	0,016	-0,032	0,016
T-6	β_6	0,025	0,012	2,092	0,036	0,002	0,048	0,002	0,048
T-7	β_7	0,576	0,011	51,288	0,000	0,554	0,598	0,554	0,598
T-8	β_8	-0,295	0,012	-25,119	0,000	-0,318	-0,272	-0,318	-0,272
T-9	β_9	-0,013	0,012	-1,087	0,277	-0,037	0,011	-0,037	0,011
T-10	β_{10}	-0,040	0,012	-3,244	0,001	-0,064	-0,016	-0,064	-0,016
T-11	β_{11}	-0,071	0,012	-5,792	0,000	-0,095	-0,047	-0,095	-0,047
T-12	β_{12}	0,006	0,012	0,497	0,619	-0,018	0,030	-0,018	0,030
T-13	β_{13}	-0,023	0,012	-1,920	0,055	-0,046	0,000	-0,046	0,000
T-14	β_{14}	-0,023	0,012	-1,920	0,050	-0,046	0,000	-0,046	0,000

În implementarea metodei SARIMA (ARIMA sezonieră), s-a constatat că utilizarea datelor orare din ultima zi oferă rezultate satisfăcătoare, și, în consecință am ales un factor sezonier (ciclic) $s_t = 24$. O abordare similară a fost implementată în lucrarea [114], unde autorii au implementat un algoritm de actualizare automat al coeficienților SARIMA la fiecare set de 24 de valori. Metoda ARIMA cu componentă sezonieră integrează factori non-sezonieri, cât și sezonieri într-un model multiplicativ. O notație scurtă pentru model este prezentată în relația (5.5) :

$$ARIMA(p, d, q) \times (P, D, Q)(S) \quad (5.5)$$

unde p = ordin AR non-sezonier, d = diferențiere non-sezonieră, q = ordin MA non-sezonier, P = ordin AR sezonier, D = diferențiere sezonieră, Q = ordin MA sezonier și S = intervalul de timp al modelului sezonier repetat.

Reprezentarea grafică a funcției de autocorelație (ACF) și a funcției parțială de autocorelație (PACF) permite determinarea componentele AR și MA ale unui model ARIMA. Datorită graficelor prezentate în Figura 5.3 și Figura 5.4 se poate observa descreșterea exponențială a valorilor semnificative PACF, motiv pentru care se utilizează valorile $p = 0$ și $q = 1$. O valoare semnificativă apare ciclic la factor de întârziere 24 în graficul ACF ceea ce sugerează faptul că ciclul seriilor de timp este de $S = 24$ (o zi în termeni de ore) și, deoarece acest decalaj este pozitiv, se utilizează $P = 1$ și $Q = 0$. Graficele ACF și PACF sunt obținute pe baza valorilor curbei de sarcină agregate. Seria de timp analizată nu este diferențiată deoarece este o serie staționară; în consecință pentru SARIMA, stabilim $d = 0$ atât pentru componenta non-sezonieră cât și pentru cea sezonieră. Metoda utilizată, așa cum se menționează în ecuațiile pentru SARIMA sunt în ordinea (1,0,1) (1,0,0) (24).

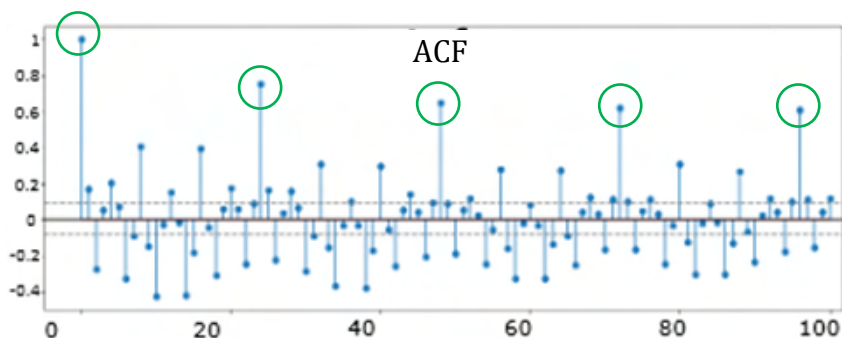


Figura 5.3 Funcția de autocorelație aplicată pe consumul agregat

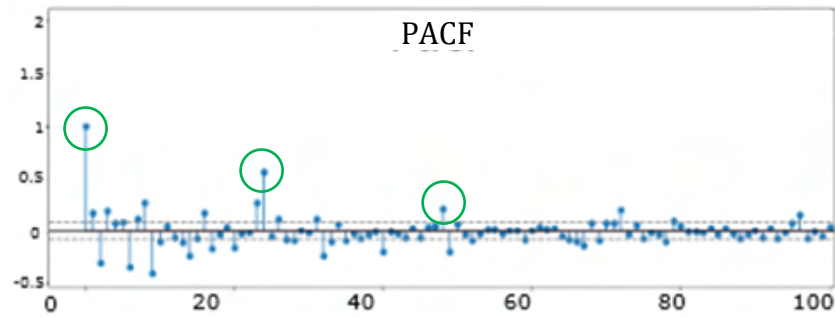


Figura 5.4 Funcția de autocorelație parțială aplicată pe consumul agregat

Conform ecuațiilor prezentate în capitolul 2 s-a implementat metoda Winter-Holts sau netezirea exponențială triplă pentru valori orare cu scopul de a realiza predicțiile pentru următoarele 24 de ore. Folosind $m = 24$ de intervale pentru fiecare zi și $t = 1..8760$, rezultă următorii factori inițiali:

$$\text{Factori sezonieri inițiali: } s_m = \frac{Y_m}{\left(\frac{y_1 + y_2 + \dots + y_{24}}{24}\right)}, m = 1, 2, \dots, 24 \quad (5.6)$$

$$\text{Factor de nivel inițiali: } \ell_0 = \frac{y_{25}}{s_1} \quad (5.7)$$

$$\text{Factor de trend inițiali: } b_0 = \frac{y_{25}}{s_1} - \frac{y_{24}}{s_{24}} \quad (5.8)$$

$$\text{Factori de nivel } \ell_t = \alpha \frac{y_t}{s_{t-m}} + (1 - \alpha)(\ell_{t-1} + b_{t-1}) \quad (5.9)$$

$$\text{Factori de trend } b_t = \beta(\ell_t - \ell_{t-1}) + (1 - \beta)b_{t-1} \quad (5.10)$$

$$\text{Factori sezonieri } s_t = \gamma \frac{y_t}{\ell_t} + (1 - \gamma)s_{t-m} \quad (5.11)$$

$$\text{Prognoză } \hat{y}_{t+k} = (\ell_t + k \cdot b_t)s_{t-M+k} \quad (5.12)$$

Urmărind relațiile (5.6) - (5.12) obținem valorile prognozate pentru următoarele 24 de ore. Coeficienții corespunzători fiecărei componente sunt $\alpha = 0,9331$, $\beta = 0,0014$, $\gamma = 1$. Aceste valori pentru coeficienți au fost obținute utilizând aplicația Solver în Excel, care sunt similare valorilor obținute prin utilizarea Python și Statsmodels v0.12.2.

Aceste metode definesc un punct de referință pentru următoarele metode de prognoză și reprezentă practici întâlnite în activitatea furnizorilor de energie electrică.

5.2. Prognoza bazată pe algoritmi cu învățare automată

Literatura este bogată în recenzii ale algoritmilor de învățare automată [115] și [116] pentru diverse aplicații, de asemenea, au fost implementate, studiate și comparate o varietate de arhitecturi [117] pentru prognoza curbelor de sarcină. Identificarea unui algoritm care să rezolve problema prognozei este dificilă, iar în cazul algoritmilor cu învățare automată nu există o abordare clară, înafară de experimentarea diferitelor variante și opțiuni de implementare.

Pentru implementarea metodelor cu învățare automată, seriile de timp sunt împărțite în mai multe secvențe care reprezintă h valori de intrare pentru fiecare caracteristică și h valori țintă, unde h reprezintă orizontul de prognoză. Instruirea rețelelor neuronale se realizează prin prelucrarea datelor sub formă matricială (tensor) prin aplicarea metodei ferestrei glisante pe seriile de timp. În primul rând, cele h valori corespunzătoare fiecărei caracteristici sunt plasate în matricea de antrenament și apoi ieșirea dorită (următoarele h ore) este aranjată într-un vector coloană cu h valori țintă.

Rețelele neuronale artificiale (RNA) sunt "antrenate" în modul în care dorim să realizăm prognoza. Pentru antrenarea rețelei, glisăm fereastra către următoarele h valori (pas = 1) atât pentru valorile de intrare X cât și pentru valorile de ieșire Y . Reiterarea acestei proceduri pentru întregul set de date de antrenare se numește învățare supravegheată.

Organizăm intrările pentru algoritmi cu învățare automată (ML) într-o matrice de antrenament $X_{antrenare}$ de dimensiune (h, X) și valorile țintă într-o matrice vector $Y_{antrenare}$ sub formă $(h, 1)$. Procedura cu fereastră glisantă este prezentată schematic în Figura 5.5. Următorul pas este să stabilim parametrii pe baza cărora se construiesc prognozele. În cazul lucrării de față se iau în considerare datele orare de consum din zilele anterioare, zile (ne)/lucrătoare, ore (ne)/lucrătoare, zile speciale, date meteorologice și diverși coeficienți autoregresivi. Diverse implementări cu acești parametrii sunt evidențiate în secțiunile de rezultate, unele abordări având rezultate mai bune față de altele.

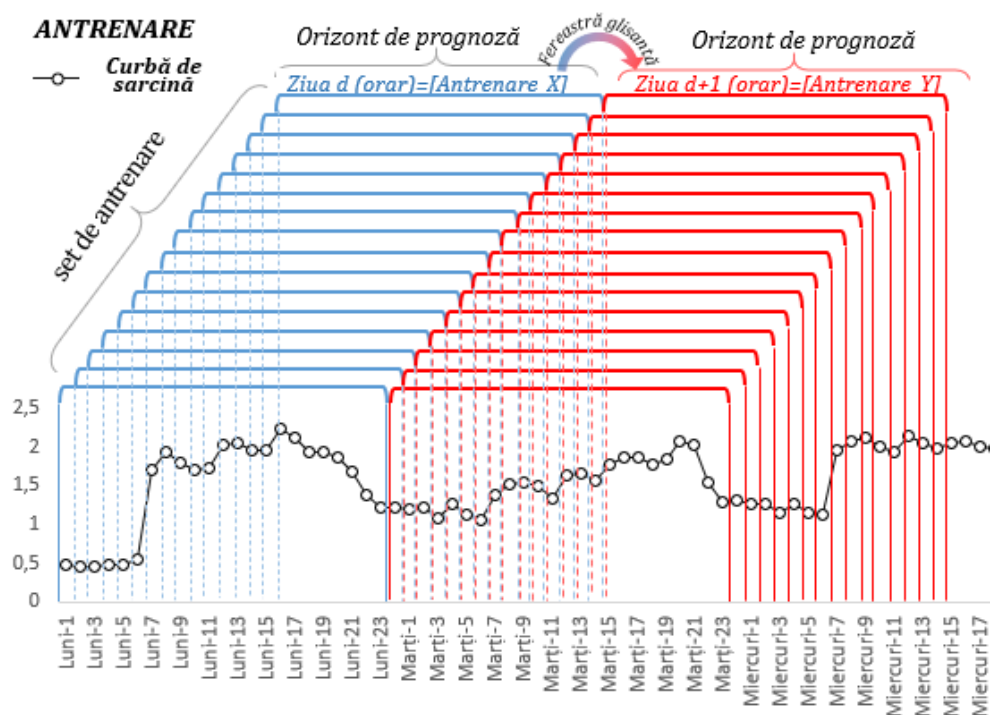


Figura 5.5 Principiul perioadei glisante [118]

După stabilirea matricelor de intrare și ieșire (țintă) se estimează o funcție f (rețea neuronală cu ponderi și erori aleatorii) care să generalizeze întregul set de date în așa fel încât să se obțină o diferență cât mai mică între valorile prognozate și cele reale. Problema prognozei seriilor de timp este definită conform relațiilor (5.13)-(5.15).

$$\hat{y} = f(x) \quad (5.13)$$

$$\hat{y}_{t+h} = f(x_t) \quad (5.14)$$

$$\hat{y}_{t+h} = f(x_{t-1}, \dots, x_{t-n}, X) \quad (5.15)$$

unde:

- h = orizont de prognoză (1, 24, 48, 168 ore);
- X = variabile care reprezintă zile (ne)lucrătoare, ziua din săptămână, ore gol/ vârf de sarcină, zile speciale, factori meteorologici;
- x_{t-n} = curbe de sarcină orare din zilele anterioare, unde n reprezintă dimensiunea istoricului luat în considerare;
- \hat{y} = valoarea prognozată;

Pentru identificarea factorilor exogeni care influențează consumul de electricitate s-au efectuat mai multe teste și analize. Motivul principal sintetizează exact diferența dintre învățare automată (ML) și algoritmi liniari, respectiv identificarea interdependențelor neliniare între două sau mai multe variabile în timp. Astfel este necesară simularea și evaluarea mai multor scenarii pe mai multe seturi de date.

Pentru realizarea prognozei cu algoritmi bazați pe învățare automată s-au identificat și implementat cinci pași elementari [37]:

Pasul 1: Definirea problemei

Definirea cu atenție a problemei necesită o înțelegere a modului în care vor fi folosite prognozele, piața de energie și beneficiarii prognozelor. Pentru acest pas s-a colaborat direct cu fiecare consumator și furnizor de energie pentru colectarea datelor, întreținerea bazelor de date și utilizarea previziunilor pentru planificările viitoare.

Pasul 2: Colectarea informațiilor și prelucrarea datelor

Pentru realizarea prognozelor sunt necesare două tipuri de informații: (a) date statistice și (b) expertiza acumulată a specialiștilor. Construirea unui model statistic consistent implică date istorice corecte și suficiente. În această teză datele de consum sunt curbe de sarcină orare și nu necesită prelucrare. În schimb, datele meteorologice trebuie să fie filtrate, interpolate și în anumite cazuri extrapolate din cauza lipsei anumitor perioade.

Pasul 3: Analiza preliminară (exploratorie)

Acest pas presupune înțelegerea curbelor de sarcină prin reprezentarea grafică comparativă a datelor orare. Stabilirea variabilelor exogene, tiparele de consum, identificarea trend-ului sau a sezonaliității sunt necesare. Analiza datelor anormale în seriile de timp trebuie explicate de personalul tehnic.

Pasul 4: Alegerea și implementarea modelelor

Alegerea celui mai bun model se realizează după compararea rezultatelor de la mai multe metode matematice, de la cele mai simple până la cele consacrate în literatură [28]. Fiecare model reprezintă un construct artificial care se bazează pe un set de ipoteze (explicite și implicite) și implică de obicei unul sau mai mulți parametri care trebuie evaluați folosind datele istorice cunoscute.

Pasul 5: Utilizarea și evaluarea unui model de prognoză.

Există multe probleme și obstacole în utilizarea și implementarea prognozelor în practică, cel mai mare obstacol fiind expunerea la risc. Metodele de evaluare se bazează pe calculul indicatorilor statistici, iar în prezenta teză se abordează evaluarea riscului din punct de vedere al pieței de energie electrică (capitolul 8).

5.2.1. Random Forest

Random Forest (RF) este un algoritm cu învățare supravegheată care se bazează pe metoda de învățare în ansamblu cu arbori de decizie propus de [119]. RF este o tehnică de agregare a datelor (bagging sau agregare bootstrap) prin care toate calculele sunt executate în paralel fără interacțiune între arborii de decizie în momentul construirii lor. RF poate fi utilizat pentru a rezolva atât sarcini de clasificare, cât și sarcini de regresie.

Numele de RF provine din ideea de agregare aleatoare a datelor și construirea unor seturi de arbori de decizie. Metoda RF este un algoritm ML eficient care ajută la îmbunătățirea unor dezavantaje aferente metodelor bazate pe arbori decizionali singulari [120].

Arborii de decizie învață cum să împartă cel mai bine setul de date în subseturi pentru a prezice valoarea țintă utilizând un criteriu de decizie conform pașilor următori:

Pas 1: Algoritm selectează probe aleatorii din setul de date furnizat.

Pas 2: Algoritm va crea un arbore de decizie pentru fiecare eșantion selectat, apoi va obține un rezultat de predicție din fiecare arbore de decizie creat.

Pas 3: "Votarea" va fi apoi efectuată pentru fiecare predicție. Pentru o problemă de clasificare, se folosește impuritatea Gini (5.16) sau Entropia (5.17), iar pentru o problemă de regresie, se folosesc ecuațiile (5.18) sau (5.19).

Pas 4: În cele din urmă, algoritmul va selecta cel mai "votat" rezultat al predicției ca predicție finală.

Pentru clasificare criteriul de decizie este impuritatea Gini sau criteriul de dublare (entropie) [121], iar pentru regresie se folosește reducerea varianței folosind metoda celor mai mici pătrate (eroare pătrată medie) conform Tabel 5.3.

Tabel 5.3 Criterii de decizie pentru RF

Impuritate Gini	Clasificare	$\sum_{i=1}^C f_i(1 - f_i)$	f_i este frecvența etichetei i la un nod, C este numărul etichetelor unice.	(5.16)
Entropie	Clasificare	$\sum_{i=1}^C -f_i \log(f_i)$	f_i este frecvența etichetei i la un nod, C este numărul etichetelor unice.	(5.17)
Variație / MSE	Regresie	$\frac{1}{m} \sum_{i=1}^n (y_i - \mu)^2$	y_i valori țintă, m este numărul de valori din setul de date, și μ este media calculată cu $\frac{1}{m} \sum_{i=1}^n y_i$.	(5.18)
Variație / MAE	Regresie	$\frac{1}{m} \sum_{i=1}^n y_i - \mu $	y_i valori țintă, m este numărul de valori din setul de date, și μ este media calculată cu $\frac{1}{m} \sum_{i=1}^n y_i$.	(5.19)

În Figura 5.6 se ilustrează procesul de învățare pentru metoda RF utilizată în prognoză. În procesul de antrenare fiecare arbore dintr-o "pădure aleatoare" învață dintr-un eșantion aleatoriu de valori. Valorile sunt extrase cu înlocuire (bootstrapping), astfel antrenarea aceluiași arbore pe valori diferite, va rezulta într-o variație mare în raport cu un anumit set de date de antrenament, dar "pădurea" va avea o variație mai mică. Condiția de împărțire este reprezentată ca o „frunză” (nod) și posibilele rezultate ca „ramuri” (margini). Acest proces de împărțire continuă până când se îndeplinește o regulă prestabilită (adâncimea maximă a arborului). Algoritmul creează o succesiune de arbori binari - fiecare nod are exact două ieșiri - găsind cea mai bună caracteristică numerică pentru a separa datele.

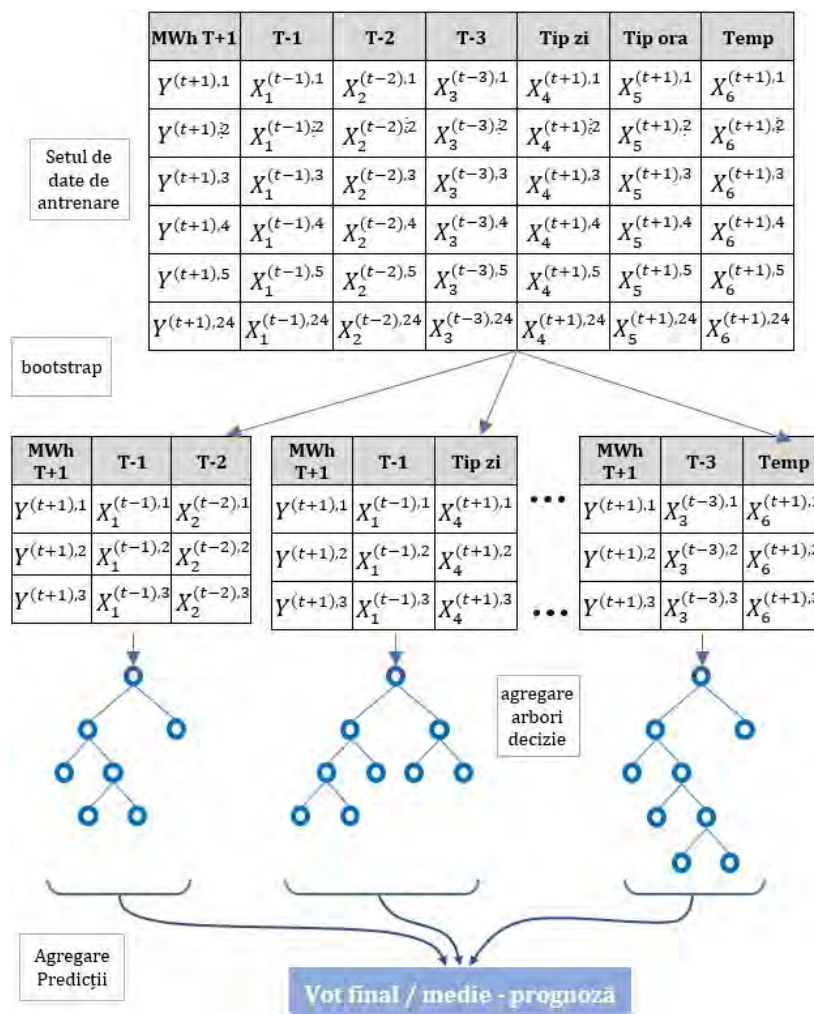


Figura 5.6 Algoritmul RF pentru prognoza [122]

RF construiesc arbori de decizie individuali în stadiul de antrenament. Estimările din toți arborii sunt apoi agregate pentru prognoza finală; pentru regresie se folosește valoarea medie obținută pentru toți arborii. Având în vedere că se utilizează mai multe rezultate pentru a lua o decizie finală, acestea sunt denumite tehnici agregate [123]. Dimensiunile “pădurii” generate pe baza curbelor de sarcină este foarte mare (imposibil de vizualizat într-o imagine), de aceea se prezintă în Figura 5.7 o parte din structura RF.

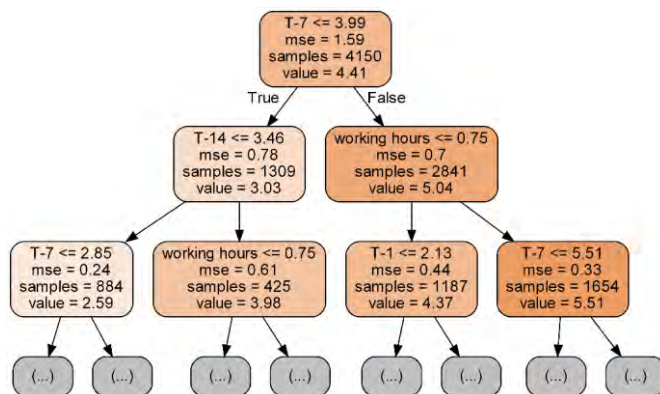


Figura 5.7 RF cu doi arbori decizionali

Implementarea căutării aleatorii permite restrângerea intervalului pentru fiecare variabilă folosită în prognoză. Astfel căutarea se concentrează în mod explicit pe fiecare combinație de setări posibile. Această metodă se realizează în Python cu metoda GridSearchCV¹², o metodă care, în loc de eşantionare aleatorie dintr-o distribuție, evaluează toate combinațiile definite în Tabel 5.4. Pentru a utiliza Grid Search, se realizează o altă grilă bazată pe cele mai bune valori oferite de căutarea aleatorie.

Tabel 5.4 Configurarea RF

Parametri folosiți	Parametrii analizați	Detalii despre hiperparametrii utilizați
{'bootstrap': True, 'ccp_alpha': 0.0, 'criterion': 'mse', 'max_depth': None, 'max_features': 'auto', 'max_leaf_nodes': None, 'max_samples': None, 'min_impurity_decrease': 0.0, 'min_impurity_split': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'min_weight_fraction_leaf': 0.0, 'n_estimators': 100, 'n_jobs': None, 'oob_score': False, 'random_state': None, 'verbose': 0, 'warm_start': False}	{'bootstrap': [True, False], 'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, None], 'max_features': ['auto', 'sqrt'], 'min_samples_leaf': [1, 2, 4], 'min_samples_split': [2, 5, 10], 'n_estimators': [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000]}	n_estimators = numărul de arbori din pădure max_features = numărul maxim de caracteristici luate în considerare pentru divizarea unui nod max_depth = numărul maxim de niveluri din fiecare arbore de decizie min_samples_split = numărul minim de puncte de date plasate într-un nod înainte ca nodul să fie divizat min_samples_leaf = numărul minim de puncte de date permise într-un nod frunză bootstrap = metoda de eşantionare a punctelor de date (cu sau fără înlocuire)

Pentru evaluarea prognozelor realizate cu RF, adiacent indicilor prezentați în secțiunea *Evaluare prognoză* se utilizează valoarea lui R^2 descrisă de ecuațiile (5.20) - (5.22). Acest coeficient indică gradul de dependență al variabilei predictibile de variabilele independente sau exogene.

¹² scikit-learn.org/stable/generated/sklearn.model_selection.RandomizedSearchCV.html

Coeficient de determinare $R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$ (5.20)

Suma reziduurilor pătrate $SS_{res} = \sum_i (y_i - \hat{y}_i)^2$ (5.21)

Suma reziduurilor pătrate totale $SS_{tot} = \sum_i (y_i - \bar{y})^2, \bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$ (5.22)

În [124] se afirmă că coeficientul de determinare (R^2) este mai potrivit decât SMAPE, MAE, MAPE, MSE și RMSE în evaluarea analizei de regresie. Pentru datele de antrenare se obține un scor $R^2 = 0,992$, iar pentru testare se obține $R^2 = 0,8905$. Coeficientul R^2 este o măsură statistică care indică cât de bine sunt efectuate prognozele în raport cu valorile reale. Un coeficient cu valoarea 1 indică faptul că prognozele se potrivesc perfect pe datele cunoscute.

5.2.2. Rețele neuronale artificiale

Există multe tipuri de rețele neuronale care pot fi clasificate în funcție de: structură, flux de date, neuroni utilizați și densitatea lor, straturi, adâncimea staturilor și filtre de activare [125]. În funcție de structură se deosebesc două categorii importante ilustrate în Figura 5.8.

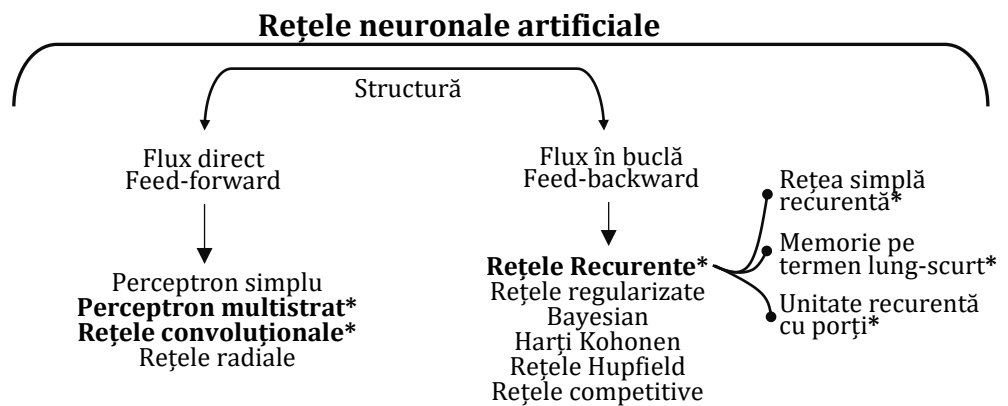


Figura 5.8 Clasificarea rețelelor neuronale artificiale¹³ [126]

¹³ Algoritmii notați cu * au fost implementați în această lucrare, sub formă singulară și combinații. Spre exemplu s-au implementat rețele CNN cu LSTM sau LSTM cu GRU.

Rețelele neuronale artificiale feed-forward, numite deseori perceptroni multistrat (MLP), sunt metode de învățare automată. Scopul unei rețele feed-forward este de a aproxima o funcție $f: \mathbb{R}^n \rightarrow \mathbb{R}$ între valorile de intrare și cele de ieșire. În cazul seriilor de timp, ne referim la o problemă de regresie, unde $Y = f(X, \theta)$. Funcția corelează o intrare X la o valoare țintă Y , pentru ca ulterior pe baza valorilor X (reale) să fie construită prognoza în orizontul de timp dorit. O rețea neuronală feed-forward definește o relație $Y = f(X, \theta)$ și "învață" valoarea coeficienților θ care au obținut cea mai bună aproximare a funcției.

Modelele numite feed-forward transmit informația de intrare X prin calculele intermediare (straturile ascunse) către ieșirea Y . Nu există conexiuni de feedback în care ieșirea modelului să fie readusă la intrare. Când rețelele neuronale feedforward sunt extinse pentru a include conexiuni de feedback, acestea se numesc rețele neuronale recurente. Se poate crea o confuzie cu algoritmul de antrenare backpropagation dar acesta nu are legătură cu structura rețelei, ci mai degrabă implică modul în care sunt actualizate ponderile.

Elementele principale al rețelelor neuronale artificiale sunt neuronii artificiali (perceptronii). Stratul de intrare este format din mulți neuroni interconectați între ei în stratul ascuns. Neuronii din stratul ascuns sunt conectați la fiecare neuron din stratul de ieșire. Fiecare neuron primește intrări de la alți câțiva neuroni care se înmulțesc cu ponderile și erorile aleatorii atribuite, care apoi sunt adunate și transmise unuia sau mai multor neuroni. Neuronii artificiali, de obicei, reprezintă o funcție de activare (funcție prag) la ieșire înainte de a trece rezultatul la următoarea variabilă [127]:

$$ieșire = \begin{cases} 0, & \text{dacă } \sum_{j=1}^n w_j X_j + b \leq prag \\ 1, & \text{dacă } \sum_{j=1}^n w_j X_j + b > prag \end{cases} \quad (5.23)$$

Conexiunile sunt asociate cu ponderi (w) și erori aleatorii (b), care sunt ajustate în etapa de învățare pentru a obține rezultatul dorit (Ec. (5.24), în așa fel încât funcția cost al întregului model să fie minimizată. Cea mai simplă structură pentru rețele neuronale este prezentată în Figura 5.9.

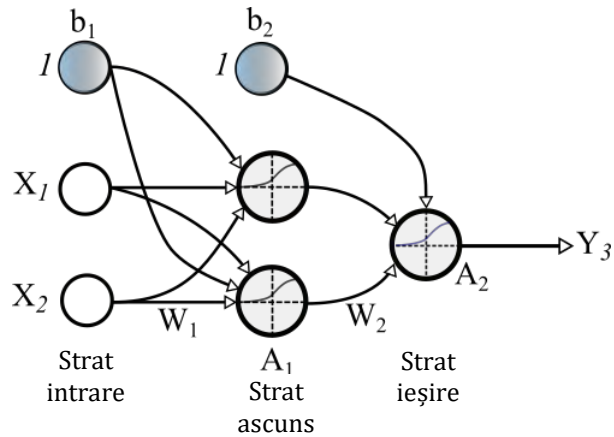


Figura 5.9 Structura unei rețele neuronale artificiale [128]

$$\begin{aligned} ieșire_1 &= \sigma(W_1 \cdot X + b_1) \\ ieșire_2 &= \sigma(W_2 \cdot A_1 + b_2) \end{aligned} \quad (5.24)$$

unde X = intrări; W_1, W_2, b_1, b_2 = ponderi și erori aleatorii; $ieșire_1, ieșire_2$ = ieșirile din fiecare strat; σ - funcția de activare sigmoid; A_1 - ieșirea din stratul ascuns. Eroarea aleatorie este o măsură care indică cât de ușoară este activarea neuronului. Pentru un neuron cu o eroare aleatorie mare, este foarte ușor să se obțină valoarea 1, dar dacă eroarea aleatorie este foarte mică, atunci este dificil să se obțină valoarea pentru activare. Pentru exemplificare se consideră ecuația (5.25).

$$ieșire = \begin{cases} 0, & \text{dacă } w \cdot X + b \leq 0 \\ 1, & \text{dacă } w \cdot X + b > 0 \end{cases} \quad (5.25)$$

Se notează cu $\sum_{j=1}^n w_j X_j$ produsul dintre valorile de intrare și ponderi. În cazul funcției de activare sigmoid $\sigma(z) = \frac{1}{1+e^{-z}}$, ieșirea unui neuron cu intrările x_1, x_2, \dots , ponderile w_1, w_2, \dots și eroarea aleatorie b devine:

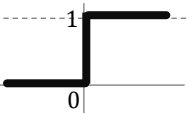
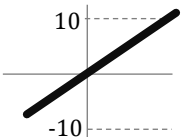
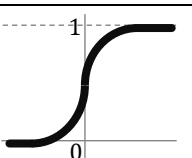
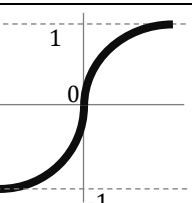
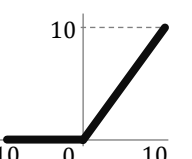
$$ieșire = \frac{1}{1 + e^{-(\sum_{j=1}^n w_j X_j + b)}} \quad (5.26)$$

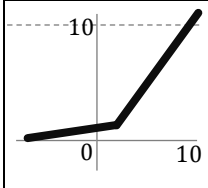
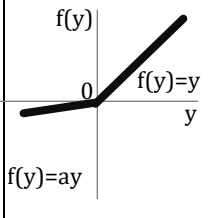
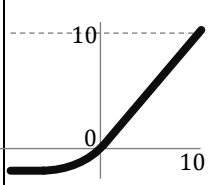
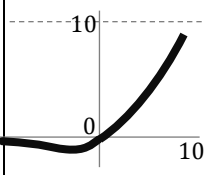
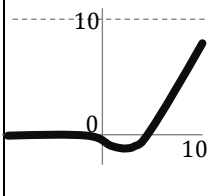
Forma funcției de activare determină dacă mici variații în ponderi Δw_j și bias Δb rezultă în variații proporționale în stratul de ieșire $\Delta ieșire$. Pentru maparea datelor de intrare la datele de ieșire este folosită următoarea formulă cu ajutorul derivatelor parțiale [129]:

$$\Delta \text{ieșire} \approx \sum_j \frac{\partial \text{ieșire}}{\partial w_j} \Delta w_j + \frac{\partial \text{ieșire}}{\partial b} \Delta b \quad (5.27)$$

unde: suma se aplică pentru toate ponderile w_j , ($\partial \text{ieșire} / w_j$) și ($\partial \text{ieșire} / \partial b$). Astfel $\Delta \text{ieșire}$ este o funcție liniară a modificărilor Δw_j și Δb în ponderi și eroarea aleatorie. Această liniaritate facilitează alegerea unor mici modificări ale ponderilor și erorilor aleatorii pentru a obține orice modificare mică dorită în ieșire. În lucrare s-au considerat, aplicat și evaluat diverse funcții de activare unde ieșirea este $f = (w \cdot x + b)$, conform Tabel 5.5. În principiu s-au aplicat funcțiile de activare care în literatura de specialitate au furnizat cele mai bune rezultate [130].

Tabel 5.5 Funcții de activare

Formă	Denumire	Formulă / Caracteristici	Ecuatie
	Funcția prag binar	$f(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$ <ul style="list-style-type: none"> - depinde de o valoare prag care decide dacă un neuron ar trebui activat sau nu. - gradientul funcției este 0, nu ajută la învățare. 	(5.28)
	Funcția liniară	$f(x) = x$ <ul style="list-style-type: none"> - activarea este proporțională cu intrarea. 	(5.29)
	Funcția sigmoid	$f(x) = \frac{1}{1 + e^{-x}}$ <ul style="list-style-type: none"> - cu cât intrarea este mai mare, cu atât valoarea de ieșire va fi mai aproape de 1. - cu cât intrarea este mai mică, cu atât ieșirea va fi mai aproape de 0. 	(5.30)
	Tangentă hiperbolică	$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ <ul style="list-style-type: none"> - similară cu sigmoid, dar diferă intervalul de ieșire. - ieșirea funcției este centrată pe zero; valorile de ieșire sunt puternic negative, neutre sau puternic pozitive. Neliniaritatea tanh este preferată în comparație cu neliniaritatea sigmoidă. 	(5.31)
	ReLU	$f(x) = \max(0, x)$ <ul style="list-style-type: none"> - funcție derivată permite backpropagarea eficientă din punct de vedere al calculului, deoarece nu activează toți neuronii simultan. Dar, dacă toate valorile negative de intrare devin zero imediat, scade capacitatea modelului de a antrena corespunzător. 	(5.32)

	Leaky ReLU	$f(x) = \max(0, 1 \cdot x, x)$ <p>- Leaky ReLU este o versiune îmbunătățită a funcției ReLU pentru a rezolva problema menționată mai sus deoarece are o mică pantă pozitivă în zona negativă.</p>	(5.33)
	Parametric ReLU	$f(x) = \max(a \cdot x, x)$ <p>- această funcție oferă o pantă părții negative a funcției ca un coeficient a. Prin efectuarea propagării înapoi, se învață cea mai potrivită valoare a lui a.</p>	(5.34)
	ELU (unitate exponențial liniară)	$f(x) = \begin{cases} x, & x \geq 0 \\ \alpha(e^x - 1), & x < 0 \end{cases}$ <p>- ReLU cu panta părții negative modificate. - ELU utilizează o curbă log pentru a defini valorile negative, spre deosebire de funcțiile leakyReLU și Parametric ReLU care au o linie dreaptă.</p>	(5.35)
	Swish	$f(x) = \frac{x}{1 + e^{-x}}$ <p>-este o funcție de activare automată. -oferă rezultate similare sau depășește ReLU pentru rețele adânci.</p>	(5.36)
	GELU (Unitate liniară de eroare gaussiană)	$f(x) \approx 0,5 * x \left(1 + \tanh \left[\sqrt{\frac{2}{\pi}} * (x + 0,044715 * x^3) \right] \right)$ <p>-neliniaritatea GELU este mai bună decât ReLU și ELU -această funcție de activare este motivată de combinarea proprietăților din dropout, zoneout și ReLU.</p>	(5.37)

Folosirea diferitelor funcții de activare vor determina valori diferite pentru ecuația (5.27). Folosind o funcție simplă ca sigmoid se simplifică partea de calcul algebric pentru derivatele parțiale.

Funcția cost

O cerință des întâlnită în învățarea automată este necesitatea unui volum mare de date pentru învățare (antrenarea rețelei), dar seturile mari de antrenament necesită multe resurse din punct de vedere al calculului (GPU, CPU și timp).

Pentru evaluarea procesului de antrenare se utilizează funcția cost care indică măsura cu care modelul estimează raportat la valorile țintă. Funcția cost utilizată de un algoritm de învățare automată se

descompune adesea ca o sumă a tuturor funcțiilor de pierdere aferente epocilor de instruire a rețelei neuronale [131]. Identificarea valorilor parametrilor rețelelor neuronale constă în optimizarea unei funcții $f(x)$ care aproximează valorile de ieșire din rețea pentru toate intrările de învățare x_i . Pentru a evalua prognozele efectuate de funcția $f(x)$ în raport cu valorile țintă y_i , se definește funcția cost, denumită și funcție pierdere sau obiectiv. O funcție de pierdere este calculată pentru o singură epocă, iar funcția cost este pierderea medie pe întregul set de date de antrenament. Cea mai utilizată funcție de cost este media pătratică descrisă de relația (5.38). O altă formulă utilizată pentru funcția cost este dată de relația (5.39) numită și eroare medie absolută (MAE). Costul MAE este util dacă datele de învățare conțin valori anormale (se manifestă prin rezultate cu valori negative și pozitive nerealiste în procesul de învățare, dar nu și în procesul de testare).

$$\text{MSE} \quad C(w, b) = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (5.38)$$

$$\text{MAE} \quad C(w, b) = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (5.39)$$

unde, w și b reprezintă totalitatea ponderilor și erorilor aleatorii din rețea, n este numărul de valori în setul de date, y este vectorul ieșirilor din rețea atunci când este introdus x , iar suma se aplică peste toate intrările de antrenament.

Ținând cont de relația (5.24) putem rescrie formula funcției cost sub următoarea formă în care valorile n , x , y sunt constante și sunt cunoscute având în vedere că fac parte din setul de date:

$$C(w, b) = \frac{1}{n} \sum_{i=1}^n (y_i - (wx_i + b))^2 \quad (5.40)$$

Calcularea derivatelor în funcție de cei doi parametri va indica cu ce factor se schimbă unul atunci când altul este modificat. În consecință, derivata lui C devine:

$$\frac{\partial C(w, b)}{\partial w} = \frac{2}{n} \sum_{i=1}^n (y_i - (wx_i + b))(-x_i) \quad (5.41)$$

$$\frac{\partial C(w, b)}{\partial b} = \frac{2}{n} \sum_{i=1}^n (y_i - (wx_i + b)) \quad (5.42)$$

Analizând forma funcției calculată cu media pătratică a erorilor, se observă faptul că rezultatul $C(w,b)$ este pozitiv, deoarece fiecare termen din sumă este pozitiv. În momentul în care $C(w,b) \approx 0$, valoarea predicției \hat{y} este aproximativ egală cu ieșirea țintă y , pentru toate intrările de antrenament x . Deci, algoritmul de antrenament a reușit să identifice ponderile, astfel încât $C(w,b) \approx 0$. În schimb, procesul de învățare nu funcționează atât de bine când $C(w,b)$ este mare - asta ar însemna că \hat{y} nu este aproape de ieșirea y pentru un număr mare de intrări. Scopul algoritmului de învățare este de a reduce la minim costul $C(w,b)$ prin identificarea valorilor ponderilor și erorilor aleatorii. Acest lucru se realizează folosind un algoritm cunoscut sub numele de metoda gradientului descendent.

Algoritmi de învățare

Folosind algoritmul de optimizare gradient descendent, ponderile sunt actualizate incremental după fiecare epocă (sau o rulare peste setul de date de învățare) cu ajutorul relațiilor (5.43)-(5.48) Mărimea și direcția actualizării ponderilor se calculează făcând un pas în sens opus gradientului funcției cost și ținând cont de rata de învățare r [129]:

$$w_{nou} = w_{initial} - r \frac{\partial C}{\partial w} \quad (5.43)$$

$$b_{nou} = b_{initial} - r \frac{\partial C}{\partial b} \quad (5.44)$$

$$\Delta w_j = -r \frac{\partial C}{\partial w_j} \quad (5.45)$$

$$\Delta b_j = -r \frac{\partial C}{\partial b} \quad (5.46)$$

Ponderile sunt apoi actualizate după fiecare epocă prin următoarea regulă de actualizare $w = w + \Delta w$, unde Δw este un vector ce conține actualizările tuturor coeficienților ponderilor w , și se calculează după cum urmează:

$$\Delta w_j = -r \frac{\partial C}{\partial w_j} = -r \sum_{i=1}^n (y_i - \hat{y}_i)(-x_j^i) = r \sum_{i=1}^n (y_i - \hat{y}_i)(x_j^i) \quad (5.47)$$

$$\Delta b_j = -r \frac{\partial C}{\partial b} = -r \sum_{i=1}^n (y_i - \hat{y}_i) = r \sum_{i=1}^n (y_i - \hat{y}_i) \quad (5.48)$$

Optimizarea prin metoda gradientului descendent reprezintă variația valorilor ponderilor pentru identificarea valorii minime pentru funcția cost. La fiecare pas se calculează panta curbei (gradientul) și rata de învățare (distanța între punctele în care se calculează gradientul). În Figura 5.10 considerăm o funcție cost cu un singur coeficient pentru ponderi.

Valorile parametrilor modelului sunt de obicei randomizate (inițiate aleator) la începutul procesului de învățare. Aceste valori aleatorii determină locația punctului pe curba de eroare (pentru un model cu un parametru), suprafața de eroare (pentru un model cu doi parametri Figura 5.11) sau hiper-planul de eroare (pentru un model cu mai mult de doi parametri).

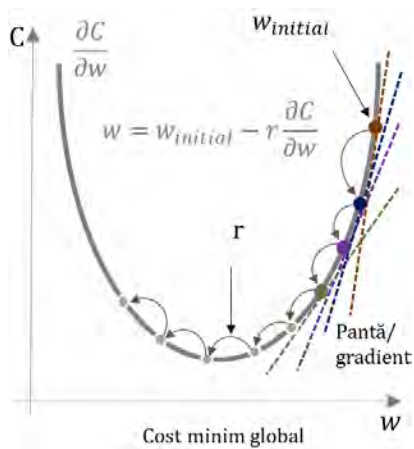


Figura 5.10 Algoritmul gradient descendent în funcție de un parametru

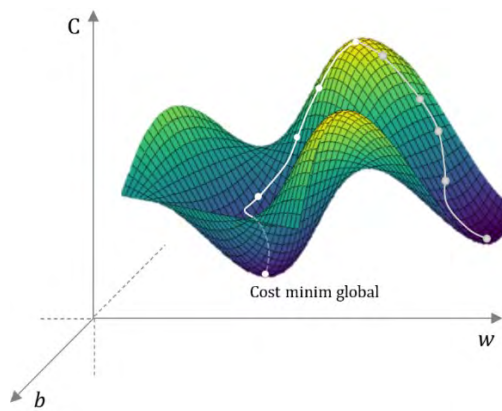


Figura 5.11 Funcția cost în funcție de doi parametri (w,b)[132]

Pentru o rețea neuronală complexă cu mai multe straturi și cu mai mulți parametri de antrenat, reprezentarea grafică este dificil de realizat și vizualizat. Mai multe informații despre posibilitățile de vizualizare a algoritmului gradient descendent (GD) sunt prezentate în [133] unde autorii detaliază importanța unei tehnici de vizualizare care oferă informații despre consecințele unei varietăți de opțiuni cu care se confruntă programatorii rețelelor neuronale, inclusiv arhitectura rețelei, selecția parametrilor de optimizare și dimensiunea lotului de antrenare.

Algoritmul backpropagation

Pentru a putea folosi un volum mare de date în învățarea automată, acestea trebuie aranjate într-o reprezentare numerică corespunzătoare implementării în limbaje de programare. De exemplu, când se dorește reprezentarea datelor de intrare ale unei rețele neuronale, avem nevoie de o modalitate de aranjare a acestora pentru a putea fi calculată funcția cost. Această reprezentare se realizează prin folosirea vectorilor, matricelor și a tensorilor conform relațiilor (5.49)-(5.51).

$$\text{Vector (Tensor 1D)} \quad \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad (5.49)$$

$$\text{Matrice (Tensor 2D)} \quad \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \quad (5.50)$$

$$\text{Tensor (Tensor 3D sau mai mare)} \quad \begin{bmatrix} [1 & 2] & [3 & 2] \\ [1 & 7] & [5 & 4] \end{bmatrix} \quad (5.51)$$

Pentru exemplificare în Tabel 5.6 se evidențiază modul în care sunt aranjate datele pentru o RNA, pentru care introducem trei valori și dorim prognozarea celei de a patra valori. Pentru realizarea prognozelor orare a consumului de energie electrică, seriile de timp sunt organizate sub forma unui tensor de intrare care cuprinde mai multe variabile, iar valorile țintă sunt aranjate sub forma unui vector care conține intervalele pentru ziua următoare sau orizontul de timp considerat pentru prognoză.

Tabel 5.6 Exemplu de reprezentare a datelor

Xantrenare	Yvalori țintă
([[[1, 2, 3],	([4],
[2, 3, 4],	[5],
[3, 4, 5]],	[6],
[[4, 5, 6],	[7],
[5, 6, 7],	[8],
[6, 7, 8]],	[9],
[[7, 8, 9],	[10],
[8, 9, 10],	[11],
[9, 10, 11]]])	[12]]

Pentru detalierea algoritmului backpropagation se folosește notația w_{jk}^l pentru ponderile aferente conexiunii de la neuronul k aflat în

stratul $(l - 1)$ către neuronul j aflat în stratul (l) . În mod similar procedăm pentru bias (b_j^l) și ieșirea din funcția de activare (o_j^l) , unde (l) reprezintă stratul și (j) neuronul din strat. Ținând cont de aceste notații și conform (5.24) se obține:

$$o_j^l = \sigma \left(\sum_k w_{jk}^l o_k^{l-1} + b_j^l \right) \quad (5.52)$$

Pentru a rescrie această expresie într-o formă matricială, definim matricea ponderilor (w^l) care cuprinde valorile w_{jk}^l și bias (b_l) cu (b_j^l) pentru fiecare strat (l) . Mai rămâne să definim un vector (o^l) pentru ieșirea care va avea valorile (o_j^l) și aplicarea funcției σ pentru fiecare element din vector $\sigma(v)_j = \sigma(v_j)$. Folosind aceste notații obținem următoarea formulă:

$$o^l = \sigma(w^l o^{l-1} + b^l) \quad (5.53)$$

Folosirea corectă a ecuației (5.52) pentru calcularea ieșirii (o^l) implică introducerea unui termen ajutător intermediar $z^l = w^l o^{l-1} + b^l$, care reprezintă intrarea ponderată în neuronii din stratul (l) . Prin urmare (5.53) devine $o^l = \sigma(z^l)$, unde z^l are valorile cuprinse în $z_j^l = \sum_k w_{jk}^l o_k^{l-1} + b_j^l$. Pentru calcularea derivatelor parțiale mai trebuie introdus un termen intermediar δ_j^l , care reprezintă eroarea de la neuronul (j) din stratul (l) și se calculează conform ecuației:

$$\delta_j^l = \frac{\partial C}{\partial z_j^l} \quad (5.54)$$

Scopul algoritmului backpropagation este de a calcula derivatele parțiale $\partial C / \partial w$ și $\partial C / \partial b$ ale funcției de cost C în raport cu orice greutate w sau bias b din rețea. Algoritmul se bazează pe operații algebrice liniare precum adunarea de vectori, înmulțirea unui vector cu o matrice sau înmulțirea a două matrici. Cele mai importante ecuații pentru algoritmul backpropagation sunt descrise de relațiile (5.55) - (5.58):

Ecuatie pentru calcularea erorii din stratul de ieșire	$\delta^L = \nabla_o C \odot \sigma'(z^L)$	(5.55)
--	--	--------

Ecuatie pentru eroarea (δ^l) în funcție de eroarea din următorul strat, (δ^{l+1})	$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$	(5.56)
Ecuatie pentru calculul ratei de modificare a costului în funcție de bias	$\delta_j^l = \frac{\partial C}{\partial b_j^l}$	(5.57)
Ecuatie pentru rata de modificare a costului în funcție de orice pondere din rețea.	$o_k^{l-1} \delta_j^l = \frac{\partial C}{\partial w_{jk}^l}$	(5.58)
$\nabla_o C$	este definit ca vectorul care conține valorile derivatelor parțiale $\frac{\partial C}{\partial a_j^l}$	
\odot	Înmulțire element cu element sau produsul Hadamard $s \odot t = s_j t_j$	

Limitările metodei GD cu algoritmul backpropagation sunt date de faptul că procesul de căutare poate ajunge într-o “regiune” cu gradient zero și este aproape imposibil de a mai ieși din această regiune indiferent de nivelul minimelor. Deși se identifică un minim într-o direcție (w), vezi Figura 5.11, există o maximă locală în altă direcție (b). Dacă conturul suprafeței este mai plat spre direcția w , algoritmul continua să oscileze și indică faptul că valorile funcție cost au converș la valori minime.

În lucrarea de față s-a implementat algoritmul gradient descendent (GD) cu optimizare Adam (Adaptare estimativă a momentului), care este o metodă care adaptează rata de învățare pentru fiecare parametru (ponderi) al rețelei neuronale. Se alege această metodă ținând cont de comparația cu alte metode (Tabel 5.7) prezentată în [134], și pentru că se obțin cele mai bune rezultate pentru erorile prognozelor.

Tabel 5.7 Variații ale algoritmului gradient descendent

Tipuri ale algoritmului gradient descendent	Algoritmi care optimizează gradientul descendent	Referință
Batch gradient descent Stochastic gradient descent Mini-batch gradient descent	Momentum	[135]
	Nesterov accelerated gradient	[136] [137]
	Adagrad	[138]
	Adadelta	[139]
	RMSprop	[140]
	Adam	[141]
	AdaMax	
	Nadam	[142]
AMSGrad	[143]	

Pentru funcția cost s-a utilizat exprimarea erorii ca medie pătratică (MSE) conform ecuației (5.38). Prin metoda gradientul descendent, eroarea se calculează pentru fiecare exemplu din setul de date de antrenament, dar numai după ce toate exemplele de antrenament au fost evaluate și modelul actualizat. Tot acest proces este ca un ciclu și se numește o “epocă de antrenament”. Dintre avantajele acestei metode amintim, eficiența computațională, generarea unui gradient de eroare stabil și o convergență stabilă. Dezavantajul cel mai mare ale metodei este faptul că gradientul poate conduce uneori la o stare de convergență care nu este cea mai bună pe care modelul o poate atinge. De asemenea, metoda necesită ca întregul set de date de antrenament să fie stocat în memorie și disponibil pentru derularea algoritmului [134].

O variantă a algoritmului de optimizare GD este gradientul descendent stocastic (SGD) calculează eroarea pentru fiecare exemplu de antrenament din setul de date, ceea ce înseamnă că actualizează parametrii pentru fiecare exemplu de antrenament unul câte unul. În funcție de problemă, acest lucru poate face SGD mai rapid decât varianta clasică a metodei. Un avantaj este că actualizările frecvente ne permit să avem o rată de îmbunătățire destul de detaliată. Actualizările frecvente, totuși, sunt mai costisitoare din punct de vedere computațional decât abordarea clasică. În plus, frecvența acelor actualizări poate determina rata de eroare să oscileze în loc să scadă în mod lent.

O altă variantă a GD este gradientul descendent mini-batch (mini-lot) care este o combinație a conceptelor SGD și gradientul descendent clasic. În această metodă se împarte setul de date de antrenament în loturi mici și se realizează o actualizare pentru fiecare lot în parte. Astfel, se creează un echilibru între robustețea metodei SGD și eficiența gradientului descendent. Dimensiunile obișnuite ale mini-loturilor variază ca orice altă tehnică de învățare automată și nu există o regulă prestabilită care să determine această dimensiune, în afară de abordarea experimentală.

Algoritmi cu învățare automată utilizați în această lucrare sunt antrenați prin metoda supreveghetă utilizând procedeul ferestrei glisante pentru prognoza pe un orizont de 24 de ore. În ecuația (5.59) se prezintă forma matriceală prin care datele sunt introduse în algoritmul de învățare automată (ML). Datele țintă sunt furnizate algoritmului sub forma matriceală prezentată în ecuația (5.60). În acest fel algoritmul este antrenat să găsească o funcție nonliniară care să mapeze corelațiile dintre intrări și ieșiri.

$$X_A = \begin{bmatrix} X_{1,ag} & X_{1,FP} & X_{1,FM} & X_{1,SM} & X_{1,FP} & X_{1,AS} & X_{1,T-n} & \cdot & X_{1,f} \\ X_{2,ag} & X_{2,FP} & X_{2,FM} & X_{2,SM} & X_{2,FP} & X_{2,AS} & X_{2,T-n} & \cdot & X_{2,f} \\ X_{3,ag} & X_{3,FP} & X_{3,FM} & X_{3,SM} & X_{3,FP} & X_{3,AS} & X_{3,T-n} & \cdot & X_{3,f} \\ X_{4,ag} & X_{4,FP} & X_{4,FM} & X_{4,SM} & X_{4,FP} & X_{4,AS} & X_{4,T-n} & \cdot & X_{4,f} \\ X_{5,ag} & X_{5,FP} & X_{5,FM} & X_{5,SM} & X_{5,FP} & X_{5,AS} & X_{5,T-n} & \cdot & X_{5,f} \\ X_{6,ag} & X_{6,FP} & X_{6,FM} & X_{6,SM} & X_{6,FP} & X_{6,AS} & X_{6,T-n} & \cdot & X_{6,f} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \cdot & \dots \\ X_{23,ag} & X_{23,FP} & X_{23,FM} & X_{23,SM} & X_{23,FP} & X_{23,AS} & X_{23,T-n} & \cdot & X_{23,f} \\ X_{24,ag} & X_{24,FP} & X_{24,FM} & X_{24,SM} & X_{24,FP} & X_{24,AS} & X_{24,T-n} & \cdot & X_{24,f} \end{bmatrix} \quad (5.59)$$

$$Y_A = \begin{bmatrix} X_{25,ag} \\ X_{26,ag} \\ X_{27,ag} \\ X_{28,ag} \\ X_{29,ag} \\ X_{30,ag} \\ \dots \\ X_{47,ag} \\ X_{48,ag} \end{bmatrix} \quad (5.60)$$

Odată ce algoritmul este antrenat, acesta este folosit pe un set de date care nu a fost folosit în procesul de antrenare. Din acest motiv datele se împart în două seturi: set de antrenare și set de testare. Setul de testare este necesar pentru validarea eficienței algoritmului.

Datele din setul de testare sunt aranjate conform relației (5.59) pentru prognozarea aferentă unui orizont de timp de 24 de ore. În această lucrare folosim patru orizonturi de timp: 1 oră, 24 ore, 48 ore și 168 ore. În Figura 5.12 se prezintă cadrul complet de implementare al prognozei (de la citirea datelor până la evaluarea prognozei) utilizând algoritmi cu învățare automată. Acest cadru de implementare este important deoarece oferă o structură clară pentru aplicarea în practică a algoritmilor și prognozelor care ulterior să fie utilizate în industrie.

Lucrarea pune accentul pe implementarea practică și ține cont de resursele avute la dispoziție atât de companiile care consumă electricitate cât și de activitatea furnizorilor. Un factor extrem de important este timpul de realizare a prognozelor, deoarece un furnizor de energie are la dispoziție puțin timp pentru a încărca prognozele în platformele de tranzacționare. Acest aspect implică ca timpul necesar antrenării algoritmului cu datele actualizate de consum și rularea

proгноzei să permită încărcarea prognozelor pe platforma informatică a OPCOM (operatorul pieței de energie electrică).

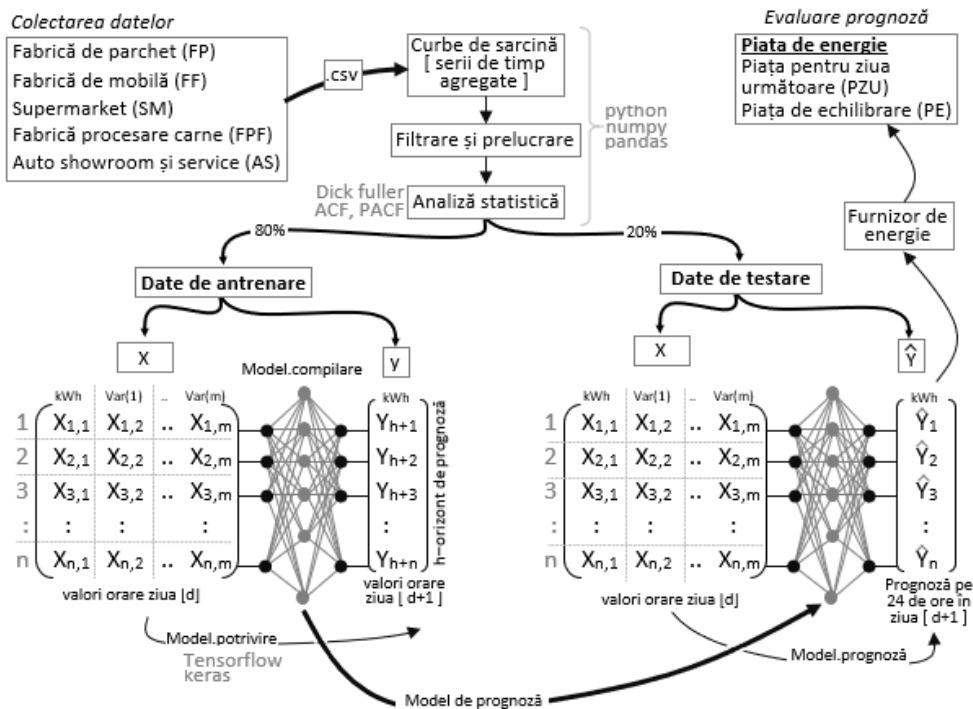


Figura 5.12 Metodologia completă de implementare a prognozei

Dezvoltarea, validarea și implementarea acestor modele de prognoză în activitatea participanților la piața de energie poate să elimine reticiența factorilor de decizie față de prognoză. Această reticență este generată în principal de complexitatea problemei, de resursele necesare pentru implementare și nu în ultimul rând de faptul că uneori rezultatele obținute pot genera pierderi financiare.

Metodele de învățare automată oferă multiple posibilități pentru prognozarea curbelor de sarcină, cum ar fi învățarea automată a dependenței temporale de factori exogeni și gestionarea automată a structurilor temporale, cum ar fi tendințele și sezonalitya.

În general, rețelele neuronale, cum ar fi algoritmi perceptron multistrat (MLP), oferă capacități neliniare ce sunt furnizați de un număr restrâns de algoritmi clasici ce sunt utilizați pentru rezolvarea problemelor de regresie. Rețelele neuronale sunt algoritmi robuști la variații în datele de intrare, iar rețeaua creată susține învățarea rapidă, respectiv prognoza. Rețelele neuronale nu trebuie să realizeze estimări

exacte cu funcția de mapare, de aceea pot cu ușurință să gestioneze relații liniare și neliniare. Pentru intrări se poate specifica un număr arbitrar de caracteristici, oferind suport direct pentru prognozele cu mai multe variabile.

Rețelele neuronale convoluționale (CNN) sunt un tip de rețele neuronale care au fost concepute pentru a gestiona în mod eficient datele sub formă vizuală. Capacitatea CNN-urilor de a învăța și extrage automat caracteristici din datele brute de intrare poate fi aplicată problemelor de prognozare a seriilor temporale. O secvență de observații poate fi tratată ca o imagine unidimensională pe care un model CNN o poate citi și cripta [144]. Rețelele CNN sunt capabile să identifice automat, să extragă și să cuantifice caracteristicile datelor de intrare. CNN-urile beneficiază de avantajele MLP-urilor pentru prognoza seriilor temporale și necesită mai puține resurse computaționale decât rețelele recurente. Modelul poate învăța o reprezentare dintr-o secvență mare de intrare care este cea mai relevantă pentru problema de prognoză.

Rețelele neuronale recurente, cum ar fi rețeaua cu memorie pe termen scurt sau LSTM, au avantajul că memorează ieșirea unui neuron prin transmiterea acesteia ca și intrare în neuronul următor. Această caracteristică nu este oferită de MLP sau CNN. Sunt un tip de rețele neuronale care adaugă suport pentru datele de intrare alcătuite din secvențe de observații. Această capacitate a LSTM-urilor a fost folosită cu succes în problemele complexe de procesare a limbajului natural, cum ar fi traducerea automată neuronală, în care modelul trebuie să învețe relațiile complexe dintre cuvinte atât într-o anumită limbă, cât și între limbi, în traducerea unei limbi în alta. Cel mai relevant context de observații de intrare la rezultatul așteptat este învățat și se poate schimba dinamic. Modelul învață atât corelarea intrărilor la ieșiri prin determinarea parametrilor rețelei (ponderi) cât și utilitatea intrărilor pentru rețea datorită buclei de feedback, astfel poate schimba dinamic contextul de învățare.

Pentru învățarea supravegheată a fost dezvoltat un program în limbajul de programare Python, folosind librăriile Keras și API-ul Tensorflow. Partea de citire a fișierelor .csv (comma separated value), aranjarea datelor, normalizarea, organizarea în matrice de învățare și testare sunt identice pentru fiecare algoritm de ML implementat. Ceea ce diferă este structura, tipul și adâncimea rețelelor utilizate pentru fiecare algoritm. Motivul principal pentru acest demers este compararea și identificarea celor mai bune modele de prognoză. Bineînțeles, există algoritmi care au rezultate mai bune sau mai slabe în funcție de setul de

date folosite, dar în cazul acestei lucrări s-a urmărit standardizarea procesului de prognoză și simplificarea acestuia. Modul de lucru pentru prognoza pentru 24 de ore este evidențiat în pseudocodul din Figura 5.13 și este o adaptare a codului propus în [145].

```

    Import Tensorflow;
    Import libraries;
    Data: citește fișier .csv
    Data: creează cadru pentru date(df)
    Data: împărțire(df)→învățare, test
    Result:  $\hat{Y}_{1..24}$  //prognoză pe orizont de timp h
    begin
        inițializează matrice[învatare];
        for i ∈ len(învatare) do
            if end < len(învatare) then
                Append( $X_{învatare}$ )
                Append( $Y_{învatare}$ )
        end
    INVATARE
        setare model ← orizontprognoz, variabile, ieșire;
        setare model ← secvențial;
        add.layers();
        model.compile(loss = mse, optimizer = adam);
        model.fit( $X_{învatare}$ ,  $Y_{învatare}$ , epoci, batch_size);
        for i ∈ len(învatare) do
             $\hat{y}_{învatare} = model.predict(învatare)$ 
        end
        Function evaluare învatare(i: train) : real
            return rmse, rms, mae, mape;
    begin
    TEST
        inițializează matrice[test];
        for i ∈ len(test) do
             $\hat{y}_{test} = model.predict(test)$ 
        end
        Function evaluare test(i: test) : real
            return rmse, rms, mae, mape;
    end

```

Figura 5.13 Pseudocod folosit pentru implementarea software

5.2.2.1. Perceptron multistrat (MLP)

MLP este un tip de rețea neuronală feedforward compusă din mai multe straturi de neuroni, fiecare având o funcție de activare [146]. În Figura 5.14 se ilustrează o structură elementară pentru MLP, unde $X = \text{valori de intrare}$; ieșirea din primul strat ascuns $A_1 = \sigma(W_1X + b)$; ieșirea din stratul n ascuns $A_n = \sigma(W_nX + b)$. Nu există o regulă concretă pentru numărul de straturi ascunse, acesta fiind ales pe baza rezultatelor obținute.

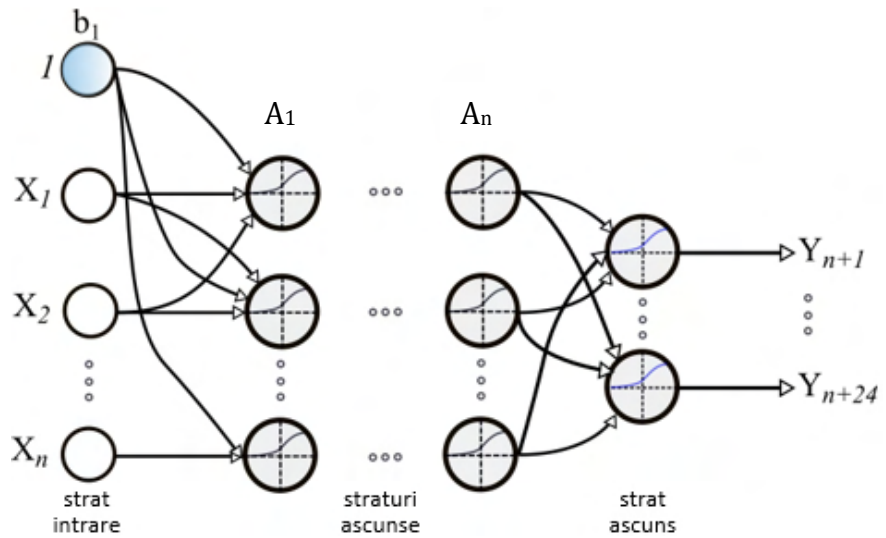


Figura 5.14 Structura perceptron multi-strat

MLP este o formă foarte populară de rețele neuronale artificiale aplicate atât problemelor de regresie, cât și de clasificare. MLP poate fi interpretat ca o mapare de funcții neliniare $f_k(X)$ de la intrări la ieșiri. Pentru realizarea prognozei se prezintă în Figura 5.15 procedul de învățare supravegheată pentru serii de timp.

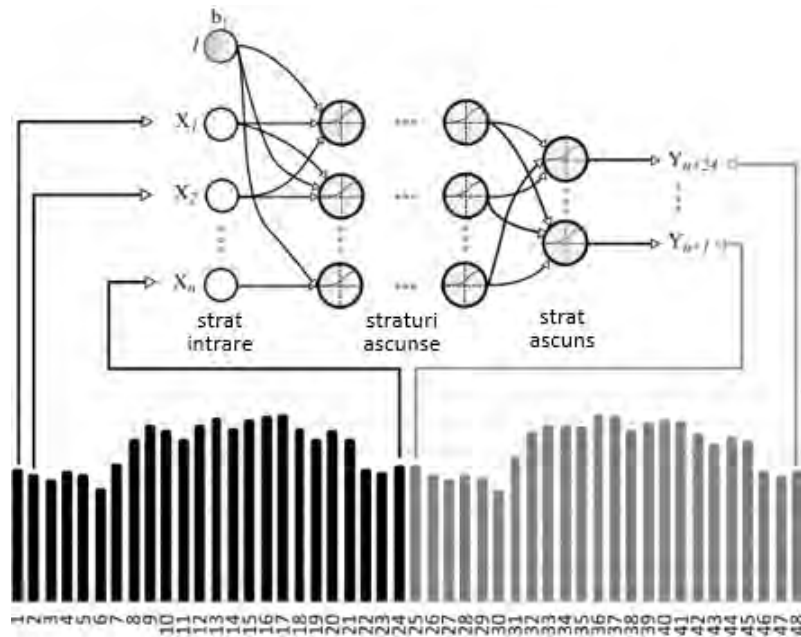


Figura 5.15 Învățarea supravegheată cu MLP

MLP constă dintr-un strat de intrare, urmat de unul sau mai multe straturi ascunse formate din neuroni cu funcții de activare și un strat de ieșire. Cele mai utilizate funcții de activare sunt $\tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}$, funcție sigmoid $\sigma(x) = \frac{1}{1+e^{-x}}$, ReLU(rectified liniar unit), Linear, Softmax, Exponential Linear Unit [147].

În această lucrare, se utilizează o arhitectură compusă din 24 de intrări, mai multe straturi ascunse cu n neuroni și 24 de neuroni în stratul de ieșire. Alegerea arhitecturii rețelei neuronale se bazează analiza rezultatelor obținute. În procesul de învățare, ponderile sunt în mod constant adaptate pe baza algoritmului gradient descent cu optimizare Adam. Studiul propus de [148] a clasat pe primul loc MLP în ceea ce privește performanța prognozării, mai bun decât SVR, RF, ARIMA și RNN. RNN se numără printre metodele mai puțin exacte bazate pe ML din acest studiu, dar autorii nu au încercat variații menite să rezolve problemele legate de gradientul care crește exponențial (“exploding”) sau scade la zero (“vanishing”).

5.2.2.2. Rețele neuronale recurente

Una dintre primele implementări ale RNN a fost propusă în [149] pentru prognoza cu o oră înainte, iar în [150] se prezintă faptul că algoritmi sunt superiori rețelelor neuronale forward. O corelație recurentă este o ecuație care definește un element al unei secvențe în funcție de elementele anterioare ale aceleiași secvențe, relația (5.61). Stările din rețeaua recurentă sunt codificările unităților ascunse, de obicei denotate ca vectori $h^{(t)}$ cu pasul de timp asociat. Secvența de stări este dependentă de stările anterioare. În plus, secvența de stare este, de asemenea, afectată de intrare la fiecare pas de timp. Astfel, relația de recurență pentru rețeaua neuronală recurentă este:

$$h^{(t)} = f(h^{(t-1)}; x^{(t)}; \theta) \quad (5.61)$$

Odată ce starea sistemului a fost dedusă în acest fel, ieșirea în orice moment este o funcție a stării ca:

$$o^{(t)} = g(h^{(t)}; \theta') \quad (5.62)$$

unde θ' este un set diferit de parametri, instruiți în mod specific pentru variabila de ieșire. Funcțiile f și g sunt aplicate la fiecare pas de timp și colectiv fiind cunoscute sub numele de celulă RNN (Figura 5.16), o unitate repetabilă în rețea.

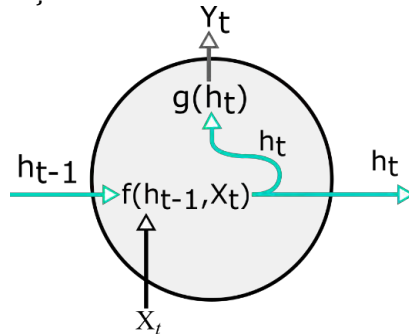


Figura 5.16 Neuron folosit în rețeaua recurentă neuronală [151]

Pentru rețelele neuronale, definirea funcțiilor înseamnă obținerea unei ieșiri prin multiplicarea vectorului ponderi la vectorul de intrare, adăugarea unui bias și apoi aplicarea funcției de activare pentru a permite modelarea neliniarității pentru ieșire. Pentru a prelua starea curentă, putem folosi o versiune simplă a funcției f .

$$h(t) = f(h^{(t-1)}; x^{(t)}; \theta) = \tanh(W h^{(t-1)} + U x^{(t)} + b) \quad (5.63)$$

În ecuația de mai sus, parametrul θ includ W , U și b . W și U sunt parametri care reprezintă matrici de ponderi, iar b este vectorul bias, deoarece, de obicei, stările sunt vectori multidimensionali. Pentru a aborda neliniaritatea, folosim tangenta hiperbolică ca și funcție de activare (tanh). Pot fi utilizate și alte funcții de activare. Ieșirea celulei RNN este:

$$o^{(t)} = g(h^{(t)}; \theta') = V h_{(t)} + c \quad (5.64)$$

unde V și c denotă ponderile și bias-ul, parametrării θ' ai funcției de ieșire g . Matricea V și vectorul c vor determina ieșirile multidimensionale. Același set de parametri este aplicat la fiecare pas de timp pentru fiecare celulă RNN.

$$h(t) = f(h^{(t-1)}; x^{(t)}; \theta) = f(h^{(t-2)}; x^{(t-1)}; x^{(t)}; \theta) = f(\dots f(h^{(0)}; x^{(1)}; \theta), \dots x^{(t)}; \theta) \quad (5.65)$$

Scrierea unei relații de recurență pentru toate variabilele de intrare este cunoscută ca desfășurarea rețelei și este prezentată în Figura 5.17. Desfășurările arată starea ascunsă în orice moment t în funcție de parametrii θ și întreaga secvență până la punctul t , X_1, \dots, X_t . Pentru prognoză, un RNN standard calculează o succesiune de ieșiri Y_{t+1}, \dots, Y_{t+m} prin iterarea ecuațiilor (5.64) și (5.65), unde m este orizontul prognozat. Pentru a antrena un RNN, trebuie să deducem θ . În timpul trecerii înainte, trebuie să calculăm prin toate stările ascunse $h(1), \dots, h(t)$ pentru a găsi gradientii parametrilor modelului. Algoritmul evaluează starea ascunsă la fiecare pas de timp și îl stochează pentru a calcula gradientii. Calculul gradientilor se realizează cu metoda backpropagation în mod secvențial în timp de la gradientul variabilelor din stratul ascuns $h(t)$ la stratul de intrare $h(1)$.

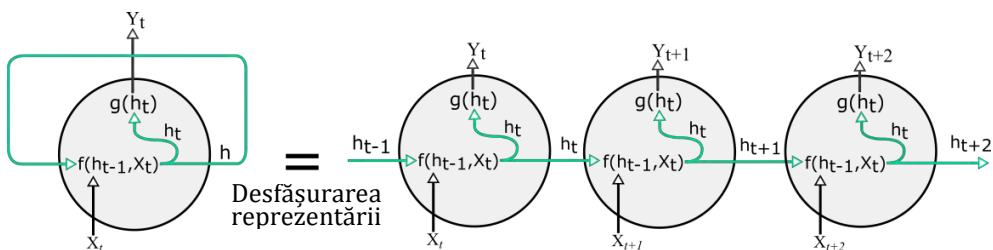


Figura 5.17 Rețele neuronale recurente – vizualizare desfășurată

În studiul nostru abordăm prognozele seriilor temporale ca învățare supravegheată. Învățarea secvențială supravegheată este descrisă de [152] și poate fi formulată după cum urmează:

Fie $(x_i, y_i)_{i=1}^N$ un set de N exemple de antrenament.

Fiecare exemplu este o pereche de secvențe (x_i, y_i) , unde $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,T_i})$ și $y_i = (y_{i,1}, y_{i,2}, \dots, y_{i,T_i})$. Provocarea este de a prezice $t + m$ elemente ale unei secvențe (y_1, \dots, y_t) .

Acest lucru poate fi extins în două moduri. În primul rând, luăm în considerare cazul în care fiecare y_t este un vector Y_t . Abordarea seriei temporale devine o problemă a mapării simultane a unui lot de serii temporale paralele: Predicție Y_{t+1} având în vedere (Y_1, \dots, Y_t) . În al doilea rând, folosim variabile exogene $(x_1, \dots, x_t, x_{t+1})$ dacă este relevant pentru Y_{t+1} .

Rețelele RNN funcționează similar cu rețelele neuronale artificiale feedforward dar procesează datele secvențial, deoarece au o memorie internă pentru a actualiza starea fiecărui neuron din rețea cu intrarea anterioară. Antrenarea rețelelor RNN cu backpropagation poate eșua din cauza dispariției sau creșterii exponențiale a gradientului,

fenomen întâlnit în rețelele adânci care combină mai multe straturi în arhitectură. Combinarea mai multor straturi în arhitectură și rezolvarea problemei gradientului poate îmbunătăți performanțele rețelei. Calcularea rețelelor RNN cu x_i intrări secvențiale și y_i ieșiri secvențiale se realizează prin utilizarea relațiilor (5.66) - (5.68), unde numărul de straturi este notat cu L și are valori cuprinse între 1 și n .

$$a_1(t) = b_1 + W_1 \cdot h_1^{t-1} + U_1 \cdot x^{(t)} \quad (5.66)$$

$$a_L(t) = b_L + W_L \cdot h_L^{t-1} + U_L \cdot h_L^{(t)} \quad (5.67)$$

$$y(t) = b_N + W_n \cdot h_n^{t-1} + U_n \cdot h_n^{(t)} \quad (5.68)$$

Proгноza se realizează folosind loturi de 24 de intrări din ziua precedentă, inclusiv caracteristicile exogene, care se furnizează rețelei și astfel se prognozează următoarele 24 de valori pentru fiecare oră, așa cum este prezentat în Figura 5.18. Pentru ca predicțiile să fie utile pentru furnizori, trebuie să prognozăm ziua următoare pentru a plasa oferte pentru piață pe ziua următoare și notificările pentru piața de echilibrare.

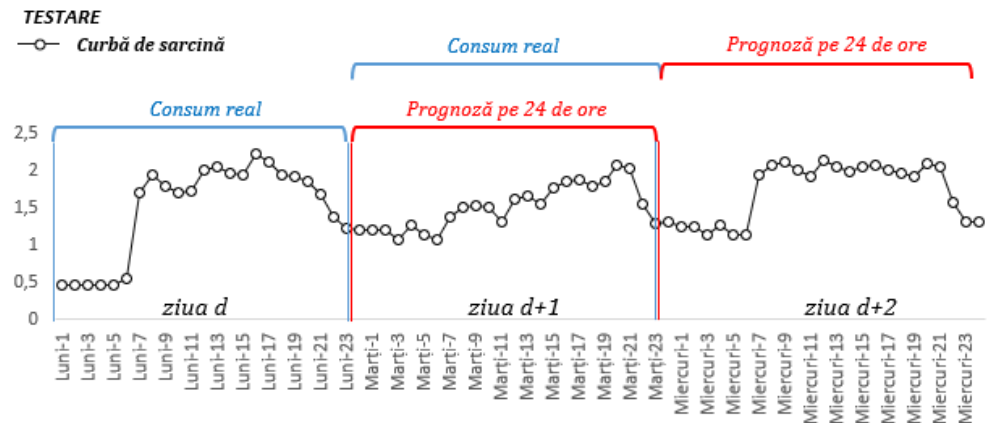


Figura 5.18 Modelul folosit pentru testare (prognoza efectivă)

5.2.2.3. Long-short term memory (LSTM)

Rețelele neuronale recurente au fost dezvoltate pentru a identifica corelații în evoluția seriilor de timp și pentru a prezice modul în care seriile temporale pot continua în viitor. Algoritmul LSTM a devenit popular cu publicarea lucrării [153] care îl prezintă ca o soluție la problema gradientului care dispare sau crește exponențial. Această problemă apare în antrenamentul rețelelor neuronale în care valoarea

gradientului utilizată pentru actualizarea ponderilor se micșorează sau crește exponențial. Conform [154] ponderile ajung să nu mai fie actualizate, iar învățarea se blochează, datorită multiplelor calcule care trebuie efectuate repetitiv.

Abordările cu LSTM pentru serii de timp au fost implementate cu succes într-o varietate de domenii [155], spre exemplu aplicații de traducere lingvistică, analiza recenziilor, chatbot, etc. Pentru prognoza curbelor de sarcină regionale lucrarea [156] prezintă LSTM ca fiind superioară față de metodele tradiționale pentru orizonturi de 24 de ore, 48 de ore, 7 zile, și 30 de zile. În lucrarea [157] prognoza cu LSTM este eficientă pentru consumul agregat rezidențial în comparație cu alți algoritmi cu învățare automat, obținând un scor MAPE de 8,18%. Modelul LSTM a fost implementat în [158] pentru prognozarea curbelor de sarcină nerezidențiale (catering, industria electronică și industria hotelieră), autorii ajungând la concluzia că prognoza fiecărui consumator nu este la fel de exactă ca și prognozele pentru consumul la nivelul stațiilor electrice (curbe de sarcină agregate). Cele mai bune rezultate s-au obținut folosind LSTM cu MAPE variind de la 15,45% la 19,57%. Pentru prognoza pe o oră a unui compresor de aer [159] prezintă un model bazat pe rețele neuronale și LSTM pentru prognoza consumului. Autorii din [160] au implementat LSTM pentru serii temporale de sarcină electrică neliniare, nestaționare și nesezoniere, cu 96 de pași înainte, cu un MAPE de 5,35%, dar fără a menționa ce tip de sarcină prognozează. În cazul unui campus universitar cu multe clădiri [161] a implementat un model AR neliniar construit cu rețele neuronale și a obținut o eroare relativă minimă de 6,6% pentru o oră înainte și o eroare maximă de 15,6% pentru prognoza pe 24 de ore.

LSTM a fost conceput pentru a compensa problema gradientului care dispare sau crește exponențial prin utilizarea unor porți prezentate în Figura 5.19 [162].

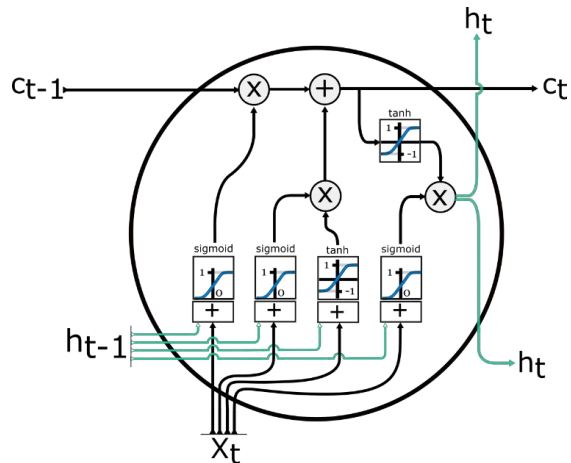


Figura 5.19 Celulă LSTM. Adaptată din [163]

În figură s-au folosit următoarele notații: X_t : vector de intrare; c_t : memoria neuronului curent; c_{t-1} : memorie din neuronul anterior; h_t : ieșirea de stare din neuronul actual; h_{t-1} : memorie din starea neuronului anterior; (\otimes) : multiplicare elementară; (\oplus) : însumare/ concatenare în funcție de elemente.

Pentru configurarea unei rețele LSTM, sunt necesari parametri prezentați în Tabel 5.8. Aceștia sunt accesibili din librăriilor Keras [92], care vor determina modul în care funcționează rețeaua:

- numărul de epoci: numărul de iterații pentru antrenare;
- funcții de activare: Sigmoid, Tangentă hiperbolică;
- structura rețelei: număr de neuroni în fiecare strat ascuns;
- optimizare GD: Keras oferă mai multe opțiuni pentru optimizarea algoritmului gradient descent (GD).

Tabel 5.8 Parametrii modelului LSTM

Parametru	Valoare
Optimizare algorithm	Adam (algoritm de optimizare a gradientului descendent)
Rată de învățare inițială	0,001
Funcții de activare	Sigmoid, Tanh
Număr de straturi ascunse	3
Număr de neuroni în fiecare strat	240/240/168
Număr de epoci	50

Conceptul de învățare supravegheat este aplicat pentru toate metodele ML implementate în această teză. Metoda Gated Recurrent Unit GRU a fost introdusă ulterior de [164] ca o alternativă mai simplă la LSTM și a devenit destul de populară. Ambele arhitecturi sunt implementate pentru prognoza consumului de energie electrică.

În modelul LSTM, există porți interne pentru stocarea informațiilor. Există cinci elemente esențiale în calculul rețelei LSTM: 1) poartă de intrare, 2) poartă de uitare, 3) poartă de ieșire, 4) celulă și 5) ieșire de stare. Figura 5.19 prezintă modelul de calcul LSTM în stratul ascuns, iar acestea sunt combinate cu alte celule pentru a forma o rețea LSTM. Operațiile cu porți, cum ar fi citirea, scrierea și ștergerea, sunt efectuate în starea de memorie a celulei. Ec. (5.69)-(5.74) prezintă reprezentarea matematică a modelului.

$$x_i = \sigma(x_t W_{x_i n} + h_{(t-1)} W_{x_i m} + b_i) \quad (5.69)$$

$$x_f = \sigma(x_t W_{x_f n} + h_{(t-1)} W_{x_f m} + b_f) \quad (5.70)$$

$$x_o = \sigma(x_t W_{x_o n} + h_{(t-1)} W_{x_o m} + b_o) \quad (5.71)$$

$$U = \text{tg}(x_t W_{U n} + h_{(t-1)} W_{U m} + b_U) \quad (5.72)$$

$$C_t = x_f \times C_{t-1} + x_i \times U \quad (5.73)$$

$$h_t = x_o \times \text{tg}(U) \quad (5.74)$$

unde x_i este intrarea porții de intrare, x_f este intrarea porții de uitare, x_o este intrarea porții de ieșire, U este semnalul de actualizare, C_t este valoarea de stare la momentul respectiv t și h_t este ieșirea celulei LSTM. Starea memoriei poate fi modificată prin decizia porții de intrare utilizând o funcție sigmoidă cu o stare 0 sau 1. Dacă valoarea porții de intrare este minimă și aproape de zero, nu va exista nicio modificare în memoria celulei de stare C_t . Stacked LSTM pot fi reprezentate în mai multe straturi în modelul de rețea. Mai multe straturi ascunse LSTM pot fi stivuite unele pe altele ca în Figura 5.20. Principalul motiv pentru stivuire este de a permite o complexitate mai mare a modelului. Rețele recurente cu multiple straturi ascunse funcționează mai bine decât rețelele mai puțin adânci [165] a prezentat o arhitectură complexă cu mai multe straturi care a obținut rezultat în medie cu 3,4% mai bune pentru traducerea automată folosind metoda LSTM “encoder-decoder”, iar [166] a arătat, de asemenea, rezultate îmbunătățite utilizând o arhitectură cu mai multe straturi recurente stivuite pentru RNN, asemănătoare cu rețeaua prezentată în Figura 5.20.

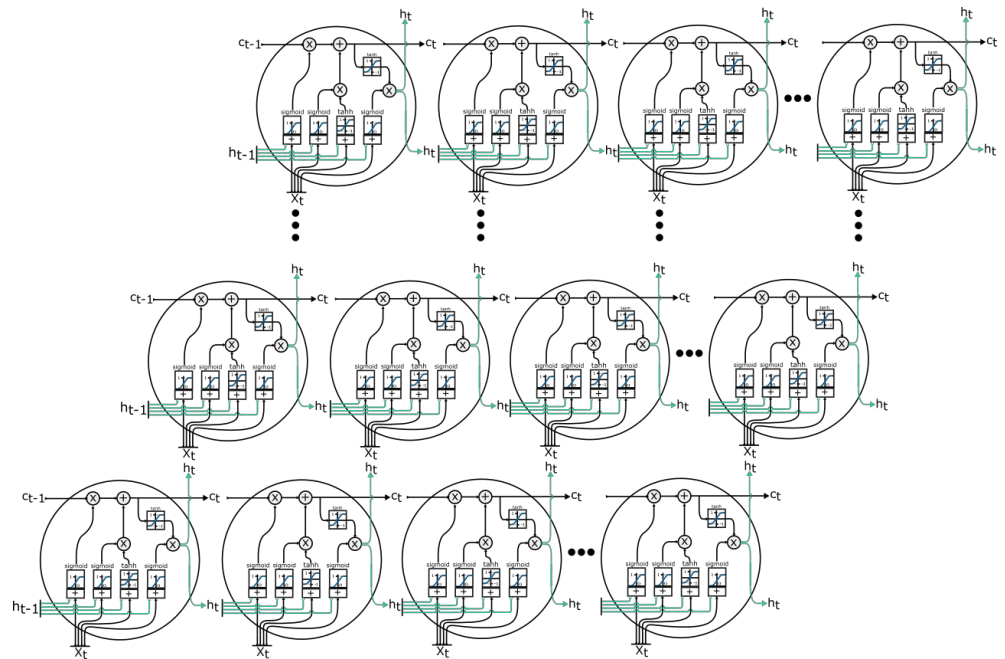


Figura 5.20 Rețea neuronală recurentă cu mai multe straturi ascunse[167]

În această lucrare se folosesc librariile Keras în care neuronul sau celula LSTM folosită în straturile rețelei neuronale este definită conform parametrilor prezentați în Tabel 5.9.

Tabel 5.9 Argumentele neuronilor GRU și Dense (folosit în stratul de ieșire)

<pre>tf.keras.layers.LSTM(units, activation="tanh", recurrent_activation="sigmoid", use_bias=True, kernel_initializer="glorot_uniform", recurrent_initializer="orthogonal", bias_initializer="zeros", unit_forget_bias=True, kernel_regularizer=None, recurrent_regularizer=None, bias_regularizer=None, activity_regularizer=None, kernel_constraint=None, recurrent_constraint=None, bias_constraint=None, dropout=0.0, recurrent_dropout=0.0, return_sequences=False, return_state=False, go_backwards=False, stateful=False, time_major=False, unroll=False, **kwargs)</pre>	<pre>tf.keras.layers.Dense(units, activation=None, use_bias=True, kernel_initializer="glorot_uniform", bias_initializer="zeros", kernel_regularizer=None, bias_regularizer=None, activity_regularizer=None, kernel_constraint=None, bias_constraint=None, **kwargs)</pre>
---	--

5.2.2.4. Gated recurrent unit (GRU)

Un alt tip de rețea neuronală recurentă este GRU, care reprezintă o versiune simplificată a celulei LSTM și este ilustrată în Figura 5.21. Algoritmul GRU este aplicat similar în această lucrare cu versiunea

implementată în [159] pe sarcini agregate la nivel regional pentru a obține o eroare MAPE de 1,13%. GRU-urile sunt mai simple decât LSTM, deoarece folosesc cu o poartă mai puțin și elimină nevoia de a face distincția între stările ascunse și celulele de memorie. Arhitectura GRU aplicată de autori în [118] examinează datele pentru consum agregat rezidențial. Rezultatele arată că RNN simplu funcționează comparabil cu GRU și LSTM atunci când este adoptat în prognozarea agregată a sarcinii. Cele mai bune rezultate sunt RMSE egal cu 13,8, iar MAE egal cu 7,5 obținute prin implementarea algoritmului LSTM. În cazul prognozei pe termen scurt [60] prezintă implementarea rețelelor GRU cu o performanță medie mai bună (2,82% MAPE) decât LSTM (3,8% MAPE) pentru sarcină de putere la nivelul unei regiuni. GRU combină poarta de intrare și poarta de uitare a LSTM într-o poartă de actualizare Z_t , iar poarta de ieșire din LSTM este numită o poartă de resetare R_t în GRU, așa cum a fost propus algoritmul inițial de către autori în articolul [168], structură ilustrată în Figura 5.21 .

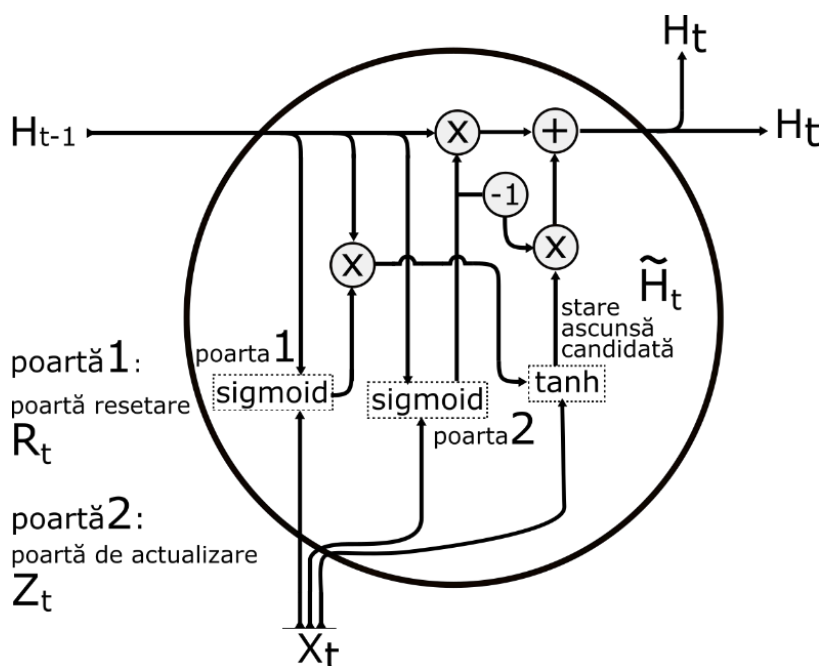


Figura 5.21 Celula GRU Gate Recurrent Unit. Adaptat din [163]

În celula GRU porțile au următoarele elemente și operații: X_t : vector de intrare; h_t : ieșire din celula curentă; h_{t-1} : memorie din celula

anterioară; (\otimes): multiplicare elementară; (\oplus): însumare / concatenare elementară.

Diferența dintre GRU-uri și RNN simplu este implementarea porții pentru starea ascunsă, care folosește această poartă pentru a determina când starea ascunsă trebuie actualizată și când să se reseteze. Intrarea este o secvență de date X_t , pentru un anumit pas de timp t , iar starea ascunsă a pasului de timp anterior este H_t (h este numărul de unități ascunse). Apoi, poarta de resetare R_t și poarta de actualizare Z_t sunt implementate după cum urmează în ecuațiile (5.75)-(5.76):

$$R_t = \sigma(X_t W_{xr} + H_{(t-1)} W_{hr} + b_r) \quad (5.75)$$

$$Z_t = \sigma(X_t W_{xz} + H_{(t-1)} W_{hz} + b_z) \quad (5.76)$$

unde W_{xr} , W_{xz} , W_{hr} și W_{hz} reprezintă parametrii ponderilor și b_r , b_z sunt erorile aleatorii. În Ecuația (5.77), poarta de resetare R_t actualizează starea ascunsă candidată \tilde{H}_t la pasul de timp t :

$$\tilde{H}_t = \tanh(X_t W_{xh} + (R_t \odot H_{(t-1)}) W_{hh} + b_h) \quad (5.77)$$

unde W_{xh} și W_{hh} sunt ponderile, b_h este eroarea aleatorie, iar simbolul \odot este operatorul înmulțire a două matrici Hadamard [169]. Pentru neliniaritatea valorilor în starea ascunsă candidată, \tilde{H}_t folosește funcția tangentă hiperbolică pentru a menține valorile în intervalul $(-1,1)$. Starea ascunsă la pasul de timp H_t , este combinația dintre stările ascunse anterioare H_{t-1} așa cum este prezentată în ecuația (5.78) și starea ascunsă candidată curentă a pasului de timp:

$$H_t = Z_t \odot H_{(t-1)} + (1 - Z_t) \odot \tilde{H}_t \quad (5.78)$$

Funcțiile de activare utilizate în celula GRU sunt tangenta sigmoidă și hiperbolică (5.79):

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \quad (5.79)$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (5.80)$$

Rețelele neuronale care sunt formate din mai mult de două straturi ascunse sunt numite rețele neuronale adânci, iar procesul de învățare se numește învățare adâncă (DL – deep learning). În această lucrare s-au implementat inclusiv rețele neuronale recurente adânci, prin urmare fiecare stare ascunsă este transmisă continuu la următorul pas de timp al stratului curent și următorul strat al pasului de timp curent. Se folosesc următoarele notații: starea ascunsă a stratului ascuns H^ℓ , variabila stratului de ieșire Y_t și funcția de activare stratului ascuns \tanh . Starea ascunsă a stratului ascuns ℓ este calculată cu ecuația (5.81) și rezultatul rețelei se obține cu ecuația (5.82):

$$H_t^\ell = \tanh(H_t^{\ell-1}W_{xh}^\ell + H_{t-1}^\ell W_{hh}^\ell + b_h^\ell) \quad (5.81)$$

$$Y_t = H_t^\ell W_{ho} + b_o \quad (5.82)$$

Unde, W_{xh}^ℓ , W_{hh}^ℓ și W_{ho} reprezintă parametrii ponderilor și b_h^ℓ , b_o sunt erorile aleatorii.

Având în vedere ecuațiile menționate, prezentăm în Figura 5.22 un cadru general de implementare al GRU pentru prognoza curbei de sarcină. Variabilele folosite în faza de antrenare sunt prezentate în figură ca variabile independente, consum anterior și o prognoză autoregresivă de ordin 9. Aceste variabile sunt selectate pe baza rezultatelor obținute, iar metoda AR(9) este utilizată pentru a evidenția posibilitatea combinării mai multor metode și s-a dorit integrarea unei variabile care netezește variațiile din zilele anterioare. În faza de testare, modelul folosește ca intrări ultimele 14 zile de consum orar, prognoza orară AR(9) pentru ziua $(d + 1)$ și variabilele exogene pentru ziua respectivă $(d + 1)$. Ieșirea rețelei reprezintă prognoza orară pentru ziua următoare (orizont de prognoză de 24 de ore).

ieșire	consum anterior orar			AR(9)	variabile independente						
Y(d+1)	X(d)	(...)	X(d-14)	AR(d+1)	W	H	Wd	Wh	T	U	Dp
Y1	X1	X1	X1	AR1	W1	H1	Wd1	Wh1	T1	U1	Dp1
Y2	X2	X2	X2	AR2	W2	H2	Wd2	Wh2	T2	U2	Dp2
Y3	X3	X3	X3	AR3	W3	H3	Wd3	Wh3	T3	U3	Dp3
Y4	X4	X4	X4	AR4	W4	H4	Wd4	Wh4	T4	U4	Dp4
...
Y22	X22	X22	X22	AR22	W22	H22	Wd22	Wh22	T22	U22	Dp22
Y23	X23	X23	X23	AR23	W23	H23	Wd23	Wh23	T23	U23	Dp23
Y24	X24	X24	X24	AR24	W24	H24	Wd24	Wh24	T24	U24	Dp24

d : zi
 l : straturi
 AR(9): autoregresie
 W : ziua din săptămână [1..7]
 H : sărbătoare legală (0,1)
 Wd : zi lucrătoare (0,1)
 Wh : oră lucrătoare (0,1)
 T : Temperatură °C
 Dp : Punct de rouă °C
 U : Umiditate %

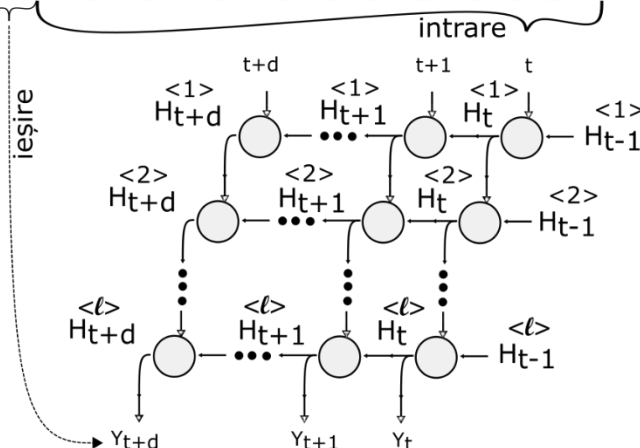


Figura 5.22 Implementarea GRU pentru prognoza curbele de sarcină [170]

Prognozele obținute pe baza setului de testare pot fi utilizate în practică de către un furnizor pentru achiziția de energie electrică de pe platformele destinate tranzacționării. Având în vedere că se prognozează următoarea zi o singură dată (24 de valori). În această lucrare, realizarea prognozelor pentru perioada de testare se realizează fără ajustarea algoritmilor cu învățare automată. Datele istorice utilizate ca intrare în rețele sunt selectate pe baza rezultatelor observate. Cele mai bune rezultate obținute folosesc ultimele două săptămâni de consum orar ca intrare în rețelele neuronale. Perioadele de întârziere mai scurte au crescut MAPE. Acest aspect înseamnă că tiparele zilnice există și se repetă săptămânal. Pentru metoda AR, decalajul a fost selectat pe baza analizei valorii p -Value. Variabilele exogene luate în considerare în prognoză au influență directă asupra consumului de energie electrică.

În cazul prognozei curbelor de sarcină pe termen scurt autorii în lucrarea [171] au folosit GRU cu o performanță medie mai bună (MAPE de 2,82%) decât LSTM (MAPE de 3,8%) pentru consumatori agregați la nivelul unei stații electrice, rezultate care sunt similare cu erorile obținute pentru algoritmi GRU și LSTM implementați în această lucrare, dar pentru curbe de sarcină ale consumatorilor non-rezidențiali. În

această lucrare se folosesc librariile Keras în care neuronul GRU este definit conform parametrilor prezentați în Tabel 5.10.

Tabel 5.10 Argumentele neuronilor GRU și Dense (folosit în stratul de ieșire)

<pre>tf.keras.layers.GRU(units, activation="tanh", recurrent_activation="sigmoid", use_bias=True, kernel_initializer="glorot_uniform", recurrent_initializer="orthogonal", bias_initializer="zeros", kernel_regularizer=None, recurrent_regularizer=None, bias_regularizer=None, activity_regularizer=None, kernel_constraint=None, recurrent_constraint=None, bias_constraint=None, dropout=0.0, recurrent_dropout=0.0, return_sequences=False, return_state=False, go_backwards=False, stateful=False, unroll=False, time_major=False, reset_after=True, **kwargs)</pre>	<pre>tf.keras.layers.Dense(units, activation=None, use_bias=True, kernel_initializer="glorot_uniform", bias_initializer="zeros", kernel_regularizer=None, bias_regularizer=None, activity_regularizer=None, kernel_constraint=None, bias_constraint=None, **kwargs)</pre>
---	--

5.2.2.5. LSTM – encoder-decoder

Metoda LSTM encoder – decoder prezentată în articolul [172] este o rețea neuronală recurentă concepută pentru a modela seriile de timp ca secvențe de date. Arhitectura acestui model cuprinde două părți: una pentru citirea secvenței de intrare și codificarea acesteia într-un vector cu lungime fixă și o a doua pentru decodarea vectorului cu lungime fixă și prognozarea secvenței de ieșire. Metoda “Encoder-Decoder” a fost introdusă cu succes în [164] și [165], pentru optimizarea proceselor de traducere automată în care numărul de intrări diferă de numărul de ieșiri. Partea “encoder” este bazată pe o rețea LSTM și mapează secvența de intrare printr-o reprezentare vectorială cu dimensionalitate fixă pentru etapa de antrenament. Partea “decoder” este o altă rețea LSTM, care folosește această reprezentare vectorială și stările ascunse din prima rețea LSTM pentru a potrivi secvența țintă după cum este prezentat în Figura 5.23.

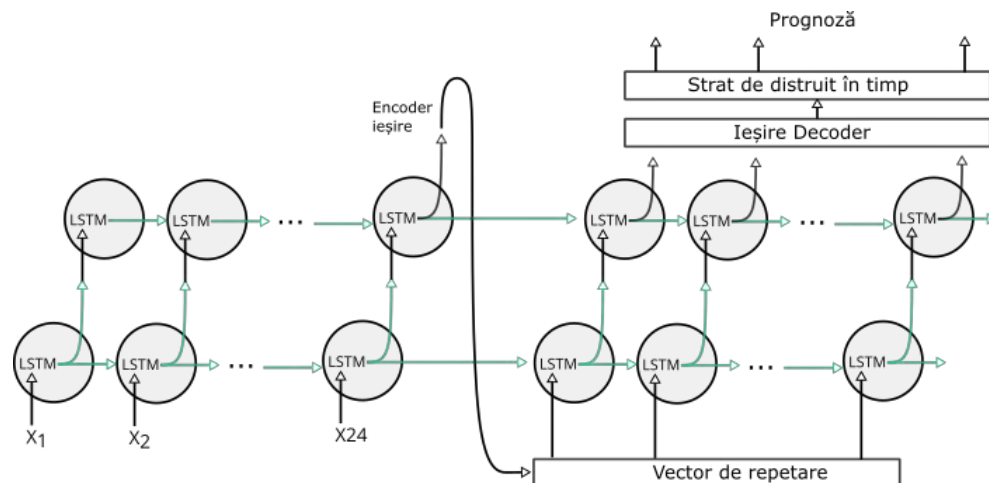


Figura 5.23 Rețea LSTM "encoder-decoder" [164]

În [173] se prezintă metoda LSTM "encoder-decoder" superioară altor metode bazate pe învățare automată pentru prognoza mai multor tipuri de serii de timp (date financiare, măsurători cu laser și radiații solare) și oferă una dintre cele mai bune performanțe pe toate seriile de timp utilizate pentru prognoza pe diferite orizonturi.

5.2.2.6. CNN - LSTM

Architectura CNN-LSTM cuprinde straturi CNN (rețele neuronale convoluționale) pentru extragerea caracteristicilor din datele de intrare care apoi sunt combinate cu straturi ascunse recurente (LSTM) pentru a folosi abilitatea de memorare a tiparelor de consum. În lucrarea [174] pentru prognoza curbelor de sarcină rezidențiale, combinația între cele două metode oferă rezultate mai bune decât LSTM și se obțin valorile MAPE de 4.01%, 4.76%, and 5.98% pentru orizonturi de 1, 3 și 6 ore. Metoda CNN-LSTM funcționează mai bine pentru probleme de clasificare, recunoaștere de imagini și video, interpretare textuală [175]. Pentru prognozarea modelelor de serii temporale, capacitatea de a extrage caracteristici temporale și spațiale permite ca stratul LSTM să memoreze în timp dependențele secvențiale (Figura 5.24). Autorii lucrării [176] indică faptul că performanța LSTM poate fi optimizată prin extragerea cu atenție a caracteristicilor în LSTM. În acest caz, straturile CNN reduc varianța seriei de timp și identifică tipare relevante.

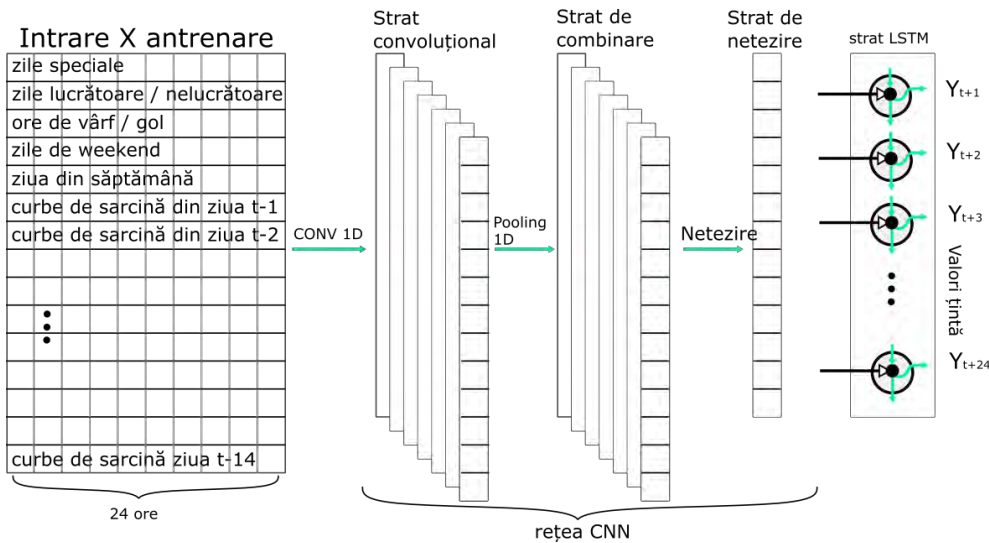


Figura 5.24 Rețea neuronală combinată CNN - LSTM [177]

În stratul convoluțional este introdus consumul anterior orar și variabile exogene, cum ar fi ziua săptămânii, zilele / orele lucrătoare sau nelucrătoare și zilele speciale. Ieșirea stratului de intrare extrage caracteristici pentru stratul CNN, care poate avea mai multe straturi ascunse. Stratul ascuns constă de obicei dintr-un strat de convoluție, un strat de activare și un strat de grupare. Stratul de convoluție aplică operația de convoluție la secvența de serii temporale multivariate primite și transmite rezultatele la următorul strat. Rezultatele prezentate în lucrarea [178] pentru prognoza calității aerului arată că LSTM și CNN-LSTM au, în general, o performanță mai bună în prognozele pe mai multe ore decât alte metode bazate pe algoritmi ML. Pentru curbe de sarcină rezidențiale, autorii în [177] au obținut cea mai bună performanță de 0,37 MSE (eroare pătrată medie) cu CNN-LSTM mai bună decât LSTM și GRU. În [179], autorii prezintă modelul CNN-LSTM superior metodei LSTM și obțin o eroare MAPE de 3,22% pentru prognoza de 24 de ore pentru consum la nivel național.

5.3. Evaluare prognoză

În evaluarea performanței metodelor utilizate pentru prognoză există mai mulți indicatori statistici care pot descrie acuratețea unui model matematic adecvat care să ofere rezultate concludente. Deseori, pentru estimări de această natură se folosesc erorile de măsură care nu

reflectă exact capacitatea tehnicilor de prognoză pentru a putea prezice cu succes observațiile viitoare.

Măsurile dependente de unitate, cum ar fi MAE (Eroarea medie absolută) și RMSE (Eroarea medie pătrată a rădăcinii) nu sunt potrivite pentru evaluare dacă seria temporală este măsurată în unități diferite. Cu toate acestea, ambele măsuri oferă informații valoroase. MAE este perfect interpretabil, deoarece încorporează abaterea medie absolută de la valorile reale. RMSE, pe de altă parte, nu este atât de ușor de interpretat, deoarece este vulnerabil la valori extreme, dar este des folosit în practică. Una dintre cele mai frecvent utilizate măsuri care evită această problemă se numește MAPE (eroare medie absolută procentuală). Aceasta rezolvă problema abordărilor menționate deoarece nu depinde de unitatea seriei temporale. În ciuda popularității sale, MAPE nu măsoară corect dacă unele dintre valorile reale sunt aproape de zero, erorile procentuale absolute corespunzătoare vor fi extrem de mari și, prin urmare, influențează rezultatul [180]. Acest motiv a dus la dezvoltarea unei forme modificate denumite SMAPE („Simetric“ MAPE), care la rândul ei are câteva dezavantaje. În această lucrare se pune accentul pe trei măsuri ale prognozei (MAPE, MAE, RMSE) și se propune definirea unei proceduri noi de cuantificare al impactului prognozei în piața de electricitate.

De obicei în evaluarea performanțelor modelului de prognoză se utilizează calculul de bază pentru diferența din estimat și real: $e(t) = \hat{y}(t) - y(t)$, unde $\hat{y}(t)$ este prognoza făcută la momentul t , iar $y(t)$ este valoarea reală la același moment t . Dacă se presupune că seria de timp conține n observații făcute pentru fiecare prognoză efectuată și există n erori de prognoză $e(t)_{t=1,2..n}$ indicatorii standard cei mai uzuali pentru evaluarea prognozei sunt:

Tabel 5.11 Formule folosite pentru evaluare prognoză

Eroarea medie (“mean error” sau “average error”):	$ME = \frac{1}{n} \cdot \sum_{t=1}^n e_t$	(5.83)
Eroarea medie absolută (“mean absolut error”):	$MAE = \frac{1}{n} \cdot \sum_{t=1}^n e_t $	(5.84)
Eroare medie pătratică (“mean squared error”)	$MSE = \frac{1}{n} \cdot \sum_{t=1}^n e_t^2$	(5.85)

Rădăcina pătrată a erorii medii pătratice ("root mean squared error"):	$RMSE = \sqrt{\frac{\sum_{t=1}^n e_t^2}{n}}$	(5.86)
Eroare medie procentual absolută ("mean average percentage error")	$MAPE = \frac{1}{n} \cdot \sum_{t=1}^n \frac{ Y_t - \hat{Y}_t }{Y_t} \cdot 100$	(5.87)
Eroare medie procentual absolută simetrică ("mean average percentage error")	$sMAPE = \frac{1}{n} \cdot \sum_{t=1}^n \frac{ Y_t - \hat{Y}_t }{\left(\frac{Y_t + \hat{Y}_t}{2}\right)} \cdot 100$	(5.88)

Eroarea medie relativă este o valoare estimată și se dorește a fi cât mai aproape de zero. Dacă eroarea medie a prognozei diferă apreciabil de zero, atunci apare un prag în prognoza indicată (o componentă continuă perturbatoare). Dacă media erorilor prognozei diferă apreciabil de zero, acest lucru poate să fie o indicație că seria de timp s-a modificat într-o anumită măsură pe care metoda utilizată nu a reușit să o sesizeze. Cei doi indicatori MAD și MSE măsoară variabilitatea erorilor de prognoză.

Calitatea unei prognoze poate fi măsurată utilizând valori sau comparând vizual valorile prezise cu valorile reale pe un orizont de prognoză. O procedură utilă este împărțirea istoricului datelor reale pentru utilizare ca eșantion de testare în care modelul va compara valorile reale și cele prognozate. În articolul [61] autorii propun un criteriu de clasificare pentru examinarea MAPE, iar în articolul [181] autorii prezintă patru intervale care caracterizează utilitatea prognozei prin clasificarea erorilor MAPE pentru consumul de electricitate. Autorii au realizat un studiu amănunțit în care analizează literatura de specialitate și prezintă percepția mediului industrial despre valorile MAPE, aceasta fiind prezentată în Tabel 5.12:

Tabel 5.12 Criteriu calitativ pentru MAPE

MAPE (%)	Acuratețea prognozei
<10	Prognoză foarte bună
10 - 20	Prognoză bună
20 - 50	Prognoză rezonabilă
>50	Prognoză slabă

O observație evidentă este că nici un model nu poate realiza o reprezentare exactă a evoluției reale a curbelor de sarcină. În această lucrare, dincolo de indicatorii prezentați, se urmărește modelarea

practică a utilizării prognozelor de către furnizorii de electricitate pentru încărcarea ofertelor de vânzare sau achiziție, pe platformele de tranzacționare ale pieței de electricitate. Astfel se poate compara în termeni financiari impactul prognozei, respectiv cuantificarea erorilor în impact financiar. Rezultatele acestei metode de evaluare se prezintă în Capitolul 8, iar pentru fiecare metodă de prognoză realizată în Capitolele 6 se determină și compară impactul financiar.

Motivele utilizării metodelor cu învățare automată în prognoză sunt necesitatea de automatizare și abilitatea de a identifica variații ale consumului când se introduc date noi în metode. În evaluarea noastră, se consideră date care nu au fost „văzute” de modele în procesul de antrenare. Perioada de testare este din perioada 15 octombrie 2019 - 20 decembrie 2019 și reprezintă 20% din întregul set de date. Folosind acest set de date împărțit se reduce riscul unor posibile probleme, cum ar fi supra-potrivirea și evaluarea greșită a rezultatelor. Împărțirea setului de date în antrenare și test este o tehnică de evaluare a performanței unui algoritm de învățare automată. Setul de date de antrenament (80%) este eșantionul de date utilizat pentru calculul parametrilor rețelei (ponderi, erori aleatorii). Setul de date de testare (20%) este eșantionul de date utilizat pentru a oferi o evaluare imparțială a unui model “învățat” pe setul de date de antrenament. Setul de date de testare confirmă dacă metoda este eficientă. În literatură nu există un procent optim de împărțire, de aceea s-a ales un procentaj de împărțire care îndeplinește obiectivul proiectului nostru, ținând cont de datele disponibile. Istoricul de date cu curbele de consum pentru toți consumatorii a fost disponibil pentru anul 2019. Datele de antrenament sunt suficiente pentru procesul de antrenament și cuprind trei sezoane, zile/ săptămâni normale de lucru, weekend-uri, vacanțe și perioade de întreținere pentru anumiți consumatori.

6. Prognoză agregată pentru consumatori

6.1 Introducere

Pentru analiza cazurilor reale din industrie s-a analizat fluxul tehnologic al consumatorilor și comportamentul furnizorilor de energie din punct de vedere al gestiunii consumului de energie și expunerea pe piața de electricitate. De obicei, un furnizor de electricitate are mulți clienți pentru care achiziționează energie. În momentul în care se încarcă ofertele bloc (preț/ cantitate) pe platforma de tranzacționare se ține cont doar de prognoza agregată a tuturor clienților. Furnizorul care deține un portofoliu mare de clienți, poate să lucreze doar cu valori agregate din istoric, dar un furnizor mic va încerca să țină legătura cu fiecare client în parte pentru a putea construi o prognoză cât mai realistă. Informațiile empirice schimbate între furnizor și consumator pot fi sintetizate, astfel:

- *păstrați productivitatea de săptămâna trecută?*
- *au apărut modificări tehnice în activitatea uzuală?*
- *în câte schimburi se lucrează până la sfârșitul săptămânii?*
- *apar schimbări majore față de productivitatea din ziua precedentă?*
- *sunt planificate revizii sau intervenții de mentenanță?*
- *păstrați același ritm de producție până la finalul lunii?*

Aceste informații oferă un anumit reper pentru prognoză, dar erorile pentru prognozele naive sunt mari în comparație cu alte metode, după cum este prezentat în capitolul cu rezultate. Pe lângă acest aspect trebuie ținut cont că delegarea unei astfel de activități unui angajat înseamnă o responsabilitate în plus și astfel pot apărea greșeli zilnice în comunicare. Un mare obstacol în această comunicare se regăsește în zilele de weekend și de sărbătoare, când angajații sunt liberi, iar consecința directă constă în lipsa de informații. Practic, în ziua de vineri se realizează prognoza pentru sâmbătă, duminică și inclusiv luni. Necesitatea prognozelor automate cu algoritmi care lucrează direct pe seriile de timp actualizate este evidentă în acest caz. Mai evidentă este pentru zilele de sărbătoare când pot să fie săptămâni întregi în care nu există contact între furnizor și consumator. Chiar dacă zilele de sărbătoare ar trebuie să fie similare, sunt greu de estimat și comportamentul consumatorului se poate schimba brusc.

Aceste informații îl pot ajuta pe furnizor să efectueze o prognoză pentru fiecare client în parte, după care va agrega aceste prognoze. De aceea în această secțiune analizăm eficiența prognozei agregate pentru consumatori. În principiu, curba de sarcină se aplatizează în momentul în care se adună mai multe curbe, și astfel variațiile față de medie scad și prognozele ar trebui să ofere erori mai mici.

6.2. Metodă și implementare

Datele istorice utilizate în această secțiune au fost prezentate pe larg în Capitolul 4 și reprezintă consumul agregat pentru întreg grupul de consumatori. Datele obținute de la consumatori sunt sub formă tabelară (.csv – comma separated value) și sunt încărcate direct în mediul de programare Jupyter Notebook. Toate instrumentele și librăriile utilizate au fost prezentate în Capitolul 4. Scopul este să se dezvolte un program care crește gradul de automatizare pentru prelucrarea datelor și aplicarea algoritmilor de prognoză. După ce datele sunt încărcate și aranjate sub formă matriceală se efectuează analize statistice și vizuale pe date. Înainte de începerea procesului de învățare se analizează necesitatea normalizării datelor. Algoritmii cu învățare automată care utilizează gradientul descendent ca tehnică de optimizare necesită normalizarea datelor dacă scala lor numerică diferă considerabil. În ecuația (5.47) se poate observa faptul că prezența valorilor caracteristice X în formulă va afecta dimensiunea cu care se iterează calculul gradientului. Pentru a asigura convergența gradientului se iterează cu intervale mici, iar pașii pentru deplasarea pe direcția de gradient sunt actualizați la aceeași viteză pentru toate caracteristicile, motiv pentru care este necesară scalarea sau normalizarea datelor înainte de a le introduce în model.

Utilizarea variabilelor la o scală similară poate ajuta algoritmul gradient descendent să convergă mai repede către minime. Normalizarea este o tehnică de scalare în care valorile sunt aranjate astfel încât să se încadreze între 0 și 1. Acest procedeu este evidențiat prin relația (6.1) cunoscut sub numele de scalare Min-Max și este implementat în Scikit-learn, un modul Python dezvoltat de autorii din lucrarea [182].

$$\text{Scalare Min-Max} \quad X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (6.1)$$

Standardizarea este o tehnică de scalare în care valorile sunt centrate în jurul mediei cu o deviație standard a unității conform relației (6.2). Aceasta înseamnă că media atributului devine zero și distribuția rezultată are o deviație standard a unității:

$$\text{Scalare Standard} \quad X_{norm} = \frac{X - \mu}{\sigma} \quad (6.2)$$

μ - media valorilor analizate;

σ - deviația standard a valorilor analizate.

Normalizarea oferă rezultate relevante atunci când distribuția datelor nu urmează o distribuție gaussiană. Standardizarea, pe de altă parte, poate fi utilă în cazurile în care datele urmează o distribuție gaussiană. De asemenea, spre deosebire de normalizare, standardizarea nu are un domeniu de limitare. Dacă există valori anormale în date, acestea nu vor fi afectate de standardizare. Cu toate acestea, alegerea utilizării normalizării sau standardizării va depinde de datele istorice utilizate și de algoritmul de învățare automată utilizat. Nu există o regulă clară și rapidă care să indice dacă datele trebuie normalizate sau standardizate. Se poate aborda această problemă prin adaptarea modelului la date brute, normalizate și standardizate și compararea performanței pentru cele mai bune rezultate. Din analizele efectuate, având în vedere că se utilizează valori în MWh pentru consum, rezultă că datele brute sunt suficiente pentru analiză, iar în momentul în care încercăm să normalizăm se obțin erori mai mari pentru prognoză. Standardizarea ridică eroarea cu 0,62% MAPE în cazul algoritmilor cu învățare automată față de datele normalizate. Utilizarea datelor normalizate oferă rezultate similare cu folosirea acestora fără prelucrare prealabilă.

În Tabel 6.1 se prezintă sintetiza tuturor metodelor implementate pentru realizarea prognozelor pe orizontul de 24 de ore aferent curbelor de sarcină agregate.

Tabel 6.1 Parametrii folosiți în construirea metodelor de prognoză

Metodă	Parametrii considerați
Naive I	Valorile orare din ziua d-7 sunt utilizate pentru prognoza orară în ziua d
Naive II	Valorile orare din ziua d-1 sunt utilizate pentru prognoza orară în ziua d
ES	Netezire exponențială triplă, 24 de perioade de timp (coeficienți utilizați $\alpha=0,9331, \beta=0,0014, \gamma=1$)
AR	Prognoză autoregresivă pentru fiecare oră se bazează pe aceeași oră din ultimele 14 zile. Coeficienții sunt prezentați în tabel

MA	Medie mobilă de ordin 3
SARIMA	Media mobilă integrată autoregresivă sezonieră (1,0,1) (1,0,0) [24]
MLP	Perceptron multistrat. Matrice de intrare [24,20]. Variabila de intrare: Ultimele 14 zile, Ziua săptămânii, Zi și ore lucrătoare/nelucrătoare, Zile speciale, temperatură. 2×straturi ascunse (300, 200). Strat de ieșire: Dens; Activare: Relu; Optimizer: Adam; Pierdere: eroare pătratică medie; Epoci: 20–100
RNN	Rețeaua neuronală recurentă. Matrice de intrare [24,20]. Variabila de intrare: Ultimele 14 zile, Ziua săptămânii, Zi și ore lucrătoare/nelucrătoare, Zile speciale, temperatură. 3×straturi ascunse (100, 100, 96). Strat de ieșire: Dens; Activare: Tanh, Sigmoid; Optimizator: Adam; Pierdere: eroare pătratică medie; Epoci: 20–100
LSTM	Memoria pe termen lung și scurt. Matrice de intrare [24,20]. Variabila de intrare: Ultimele 14 zile, Ziua săptămânii, Zile și ore lucrătoare/ nelucrătoare, Zile speciale, temperatură. 3×straturi ascunse (200, 200, 168). Strat de ieșire: Dens; Activare: Tanh, Sigmoid; Optimizator: Adam; Pierdere: eroare pătratică medie; Epoci: 20–100
LSTM encoder- decoder	Matrice de intrare [24,20]. Variabilă de intrare: Ultimele 14 zile, Ziua săptămânii, Zi și ore lucrătoare/nelucrătoare, Zile speciale, temperatură. 2 × straturi ascunse (100, 100), 1 × Repeat Vector, 1 × Strat distribuit în timp (96). Activare: Tanh, Sigmoid; Optimizator: Adam; Pierdere: eroare pătratică medie; Epoci: 20–100
GRU	Unitate recurentă închisă. Matrice de intrare [24,20]. Variabila de intrare: Ultimele 14 zile, Ziua săptămânii, Zi și ore lucrătoare/nelucrătoare, Zile speciale, temperatură. 3×straturi ascunse (200, 200, 168). Strat de ieșire: Dens; Activare: Tanh, Sigmoid; Optimizer: Adam; Pierdere: eroare pătratică medie; Epoci: 20–100
CNN- LSTM	Rețea neuronală convoluțională. Matrice de intrare [24,20]. Variabilă de intrare: Ultimele 14 zile, Ziua săptămânii, Zi și ore lucrătoare/ nelucrătoare, Zile speciale, temperatură. Straturi 2×1D Conv (24, 48); 1×MaxPooling1D strat; 1×Strat aplatizat; 2 × straturi LSTM; Strat de ieșire: dens distribuit în timp; Activare: Tanh, Sigmoid; Optimizer: Adam; Pierdere: eroare pătratică medie; Epoci: 20–100

Programul implementat pentru realizarea prognozelor este prezentat în Figura 6.1 și este structurat pentru a putea fi implementat similar pentru fiecare algoritm ML utilizat în lucrare. Se respectă împărțirea pentru seturile de date (80% învățare, 20% test), componența variabilelor exogene utilizate, cât și structura rețelelor neuronale (straturi de intrare, număr de straturi ascunse, număr de neuroni în straturile ascunse).

```

Import Tensorflow;
Import libraries;
Data: citește fișier .csv
Data: creează cadru pentru date(df)
Data: împărțire(df)→învățare, test
Result:  $\hat{Y}_{1..24}$  //proгноză pe orizont de timp h
begin
  inițializează matrice[invatare];
  for i ∈ len(invatare) do
    if end < len(invatare) then
      Append( $X_{invatare}$ )
      Append( $Y_{invatare}$ )
  setare model ← orizontprognoz, variabile, ieșire;
  setare model ← secvențial;
  add.layers();
  model.compile(loss = mse, optimizer = adam);
  model.fit( $X_{invatare}$ ,  $Y_{invatare}$ , epoci, batch, ize);
  for i ∈ len(invatare) do
     $\hat{y}_{invatare} = model.predict(invatare)$ 
  Function evaluare invatare(i: train) : real
  | return rmse, rms, mae, mape;
begin
  inițializează matrice[test];
  for i ∈ len(test) do
     $\hat{y}_{test} = model.predict(test)$ 
  Function evaluare test(i: test) : real
  | return rmse, rms, mae, mape;

```

Figura 6.1 Pseudo-cod pentru implementarea algoritmilor ML

6.3. Rezultate

Lucrarea urmărește implementarea algoritmilor de ML pentru automatizarea procesului de prognoză. Conceptul automatizării prognozei consumului de electricitate cum este prezentat de autori în articolul [183] prevede un sistem capabil să extragă informații utile din bazele de date ale unui distribuitor pentru: colectarea datelor, prelucrare, alegere algoritmi ML, antrenare algoritm de prognoză, evaluare rezultate și livrarea prognozelor. Întreg procesul menționat implică minimă intervenție din partea unui specialist.

Compararea rezultatelor este necesară pentru a identifica cea mai bună metodă pentru realizarea prognozelor de consum pentru facilitarea colaborării dintre consumator și SG. Alegerea celei mai bune metode are în vedere asigurarea următoarelor cerințe: timpul necesar realizării prognozei și capacitatea de generalizare pe setul de date de antrenare.

Timpul necesar implementării algoritmilor de prognoză este influențat de mai mulți factori:

a) Metode tradiționale

- Analiza preliminară, procesarea manuală a datelor și recalcularea tuturor parametrilor algoritmilor de prognoză la fiecare etapă.
- Cunoștințe teoretice aprofundate pentru implementarea algoritmilor.

b) Metode cu învățare automată

- Complexitatea modelelor ML influențează timpul de învățare, care este direct determinat de volumul de date utilizat, resursele hardware și software.
- Creșterea numărului de reiterări pentru etapa de învățare (număr de epoci) determină un timp mai ridicat pentru învățare și se pot obține rezultate necorespunzătoare. Respectiv, supra-adaptare (over-fitting) pe datele de învățare. Prea puține epoci determină un timp de învățare insuficient rezultând fenomenul de sub-adaptare (under-fitting).

În tabelele și graficele următoare se prezintă rezultate obținute și se evidențiază pentru fiecare algoritm implementat mai mulți indicatori pentru măsurarea erorilor. Deoarece măsura erorilor calculate și prezentate au particularități diferite prin care se cuantifică diferența dintre valoarea prognozată și cea reală, în lucrare se pune accent pe reprezentarea grafică și evaluarea financiară prezentată în capitolul 8. Se compară aceeași perioadă de timp și formulele de calcul pentru erori sunt calculate în același mod pentru fiecare algoritm utilizat. Rezultatele prezentate în acest capitol aparțin curbelor de sarcină agregate pentru clusterul de consumatori non-rezidențiali considerați și prezentați în capitolul 4.

Rezultatele prezentate sunt evidențiate atât pentru perioada de testare cât și pentru perioada de învățare, motivația fiind dată de necesitatea observării supra-adaptării sau sub-adaptării pe datele de învățare pentru algoritmi cu învățare automată. În Tabel 6.2 sunt prezentate măsurile erorilor obținute cu metodele tradiționale pentru prognoza pe 24 de ore a curbei de sarcină agregată.

Tabel 6.2 Rezultate ($h=24$) pentru prognoză agregată cu metode tradiționale

Tip		metric	Naive I	Naive II	ES	AR	MA	SARIMA
Agg - 24 ore	învățare	MAPE	10,49%	26,89%	9,40%	10,02%	10,28%	7,78%
		RMSE	0,7077	1,4385	0,5678	0,5788	0,6345	0,4901
		MAE	0,3675	0,9149	0,3568	0,3557	0,3906	0,3029
	test	MAPE	7,64%	9,72%	9,35%	6,76%	9,20%	7,56%
		RMSE	0,2009	1,8903	0,3280	0,1430	0,2944	0,1308
		MAE	0,3282	0,8921	0,4019	0,2897	0,4022	0,3089

Pentru a compara rezultate cu metodele ML se simulează similar pe structura de date de învățare și test. Coeficienții metodelor tradiționale sunt calculați în perioada de învățare, apoi aplicați pe ambele seturi de date. Dintre metodele tradiționale cea mai mică eroare (6,76% MAPE) este obținută de algoritmul autoregresiv AR (ordin 9). Se poate observa că metodele naive nu furnizează rezultate satisfăcătoare, acestea fiind cel mai des întâlnite în practica furnizorilor. Această observație vine în contextul în care pentru prognoza naivă s-a ținut cont de evoluția zilnică a consumului în perioada de testare, respectiv de tipul zilei și corespondența cu zilele trecute. Pentru metodele tradiționale erorile sunt mai mari în perioada de antrenare, deoarece este o perioadă mult mai lungă de timp în care se prognozează. În Tabel 6.3 se prezintă măsurile erorilor obținute cu algoritmi ML pentru prognoza pe 24 de ore a curbei de sarcină agregată.

Tabel 6.3 Rezultate ($h=24$) pentru prognoză agregată cu ML

Tip		metric	MLP	RNN	RF	LSTM	LSTMed	GRU	CNN LSTM
Agg 24 ore	învățare	MAPE	4,79%	4,14%	1,72%	3,96%	3,29%	3,64%	5,72%
		RMSE	0,4334	0,3995	0,1081	0,3969	0,3816	0,3895	0,4516
		MAE	0,2030	0,1825	0,07	0,1767	0,1555	0,1662	0,2397
	test	MAPE	6,00%	6,61%	5,92%	5,67%	6,23%	5,28%	6,97%
		RMSE	0,2334	0,2196	0,3839	0,1784	0,2060	0,1696	0,2444
		MAE	0,2754	0,3011	0,2743	0,2607	0,2822	0,2420	0,3308

Dintre algoritmi ML cea mai mică eroare MAPE este obținută cu metoda GRU de 5,28%. În comparație cu algoritmi tradiționali, algoritmi cu învățare automată obțin rezultate superioare. Doar metoda CNN-LSTM obține o eroare mai mare decât AR(9). Motivul pentru acest rezultat slab este dat de complexitatea rețelei care necesită mai mult timp și resurse pentru învățare. Se poate observa un scor MAPE de 5,72% pe setul de învățare, și se poate concluziona că algoritmul implementat

nu este antrenat suficient. Pentru a putea compara algoritmi s-a menținut o structură similară pentru rețelele neuronale utilizate (număr de neuroni de intrare, număr de straturi ascunse, timp de rulare, număr de epoci). În figurile următoare se prezintă grafic rezultatele fiecărui algoritm utilizat în lucrare pentru perioada de testare din 15 Octombrie 2019 până în 20 Decembrie 2019. Utilizarea algoritmilor de prognoză asupra consumului agregat oferă cele mai mici erori pentru majoritatea algoritmilor implementați în lucrare, după cum este prezentat în capitolele 7 și 8. Următoarele analize prezintă rezultatele pentru curbele de sarcină agregate pentru fiecare algoritm implementat în lucrare cu evidențierea măsurii MAPE.

Metodele naive sunt foarte simplu de implementat și sunt cel mai des utilizate în practică pentru prognoză. Chiar dacă sunt o practică utilă și simplu de realizat erorile generate sunt mari, după cum este prezentat în Figura 6.2 și Figura 6.3. Necesitatea intervenției umane introduce și un nivel de risc ridicat datorită caracterului repetitiv al activității.

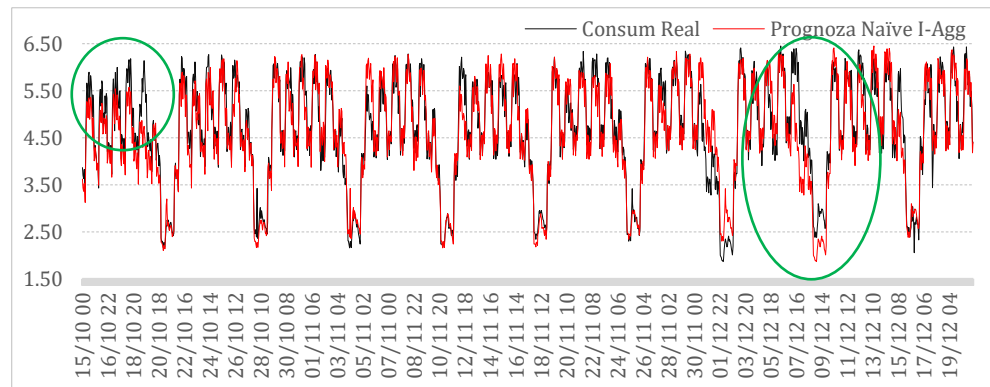


Figura 6.2 Prognoză ($h=24$) cu metoda naivă I (MAPE = 7,64%)

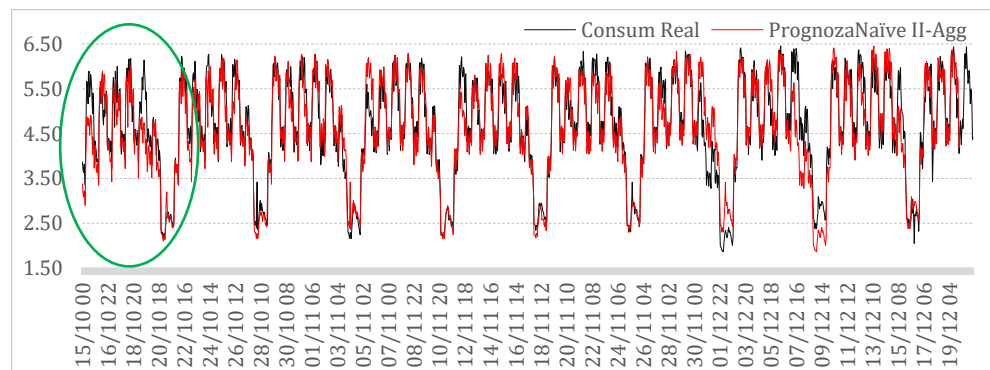


Figura 6.3 Prognoză ($h=24$) cu metoda naivă II (MAPE = 9,72%)

Conform informațiilor prezentate în capitolul 2 și 5 s-au implementat metodele tradiționale: autoregresivă, virgulă mobilă, netezire exponențială triplă și SARIMA. Prognozele orare realizate țin cont de ultimele două săptămâni pentru datele de consum, conform ecuațiilor din capitolul 5.

Netezire exponențială triplă (metoda Holt-Winters) poate fi implementată și pentru serii de timp orare, unde perioada sezonieră este $m = 24$, iar unitatea de timp adecvată este în ore, având în vedere caracterul repetitiv curbelor orare de sarcină la nivel de zi. În Figura 6.4 se poate observa că metoda Netezire Exponențială (ES) întâmpină dificultăți în zilele premergătoare zilelor de weekend și când se schimbă tiparul orar al zilei. Prognoza zilei de luni este puternic influențată de zilele de weekend. O soluție pentru rezolvarea acestei situații este împărțirea zilelor în zile lucrătoare și nelucrătoare și aplicarea metodelor de prognoză pe aceste seturi distincte de date. Această abordare nu este implementată în această lucrare deoarece se dorește compararea cu metodele ML care au ca scop principal să învețe variațiile de consum din curbele de sarcină și variabilele exogene, cu minimă intervenție în selectarea datelor care intră în algoritm.

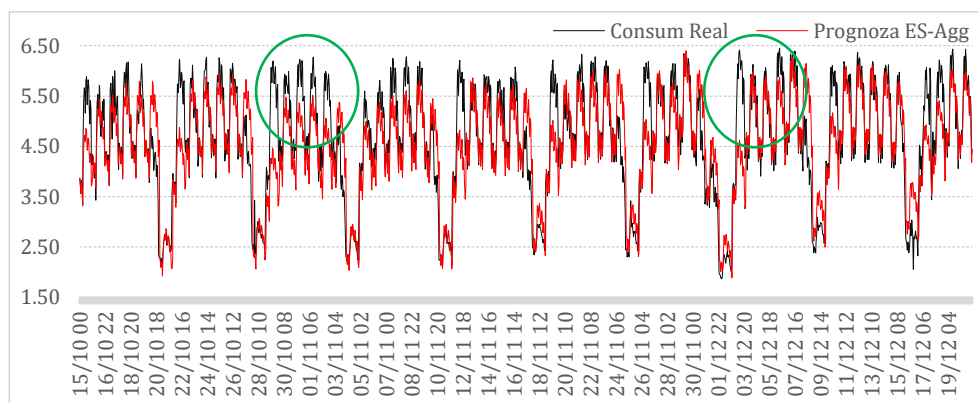


Figura 6.4 Prognoză ($h=24$) cu NE (MAPE = 9,75%)

Cel mai bun rezultat pentru metodele tradiționale a fost obținut prin implementarea algoritmului AR(9), prognoză care este realizată pe baza relațiilor autoregresive ale curbelor orare de consum anterioare și este prezentată în Figura 6.6. Respectiv, din ultimele 14 zile pe baza analizelor statistice și calcularea indicelui statistic P -value au fost

determinate 9 zile care au un impact semnificativ în consumul pe ziua următoare.

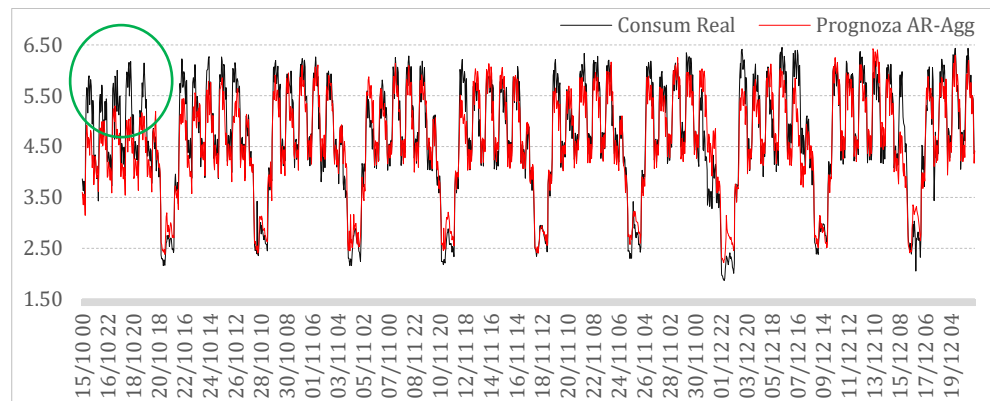


Figura 6.5 Proгноză cu metoda AR (MAPE = 6,31%)

Media mobilă (MA) implementată este un instrument simplu de analiză statistică care netezește datele de consum prin crearea unei valori medii pe ultimele trei zile. Fiind actualizată constant se poate determina evoluția trendului din setul de date. Eroarea obținută este superioară metodelor naive dar în raport cu alte metode este nesatisfăcătoare, după cum se poate observa în Figura 6.6.

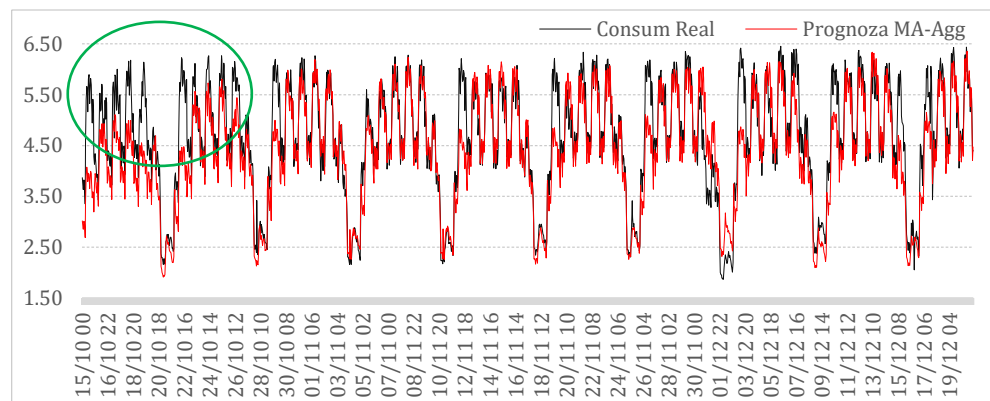


Figura 6.6 Proгноză cu metoda MA (MAPE = 8,24%)

Metoda SARIMA este prezentată în Figura 6.7 obținând o eroare MAPE de 7,56%, peste valoarea obținută de AR ceea ce indică faptul că zilele diferite ale săptămânii au adesea modele orare diferite. În special în jurul zilelor de sărbătoare sau zile libere legale, astfel încât pentru a

face o prognoză pentru o anumită oră a zilei, este importantă adaptarea zilei de prognoză la trendul zilnic al consumului.

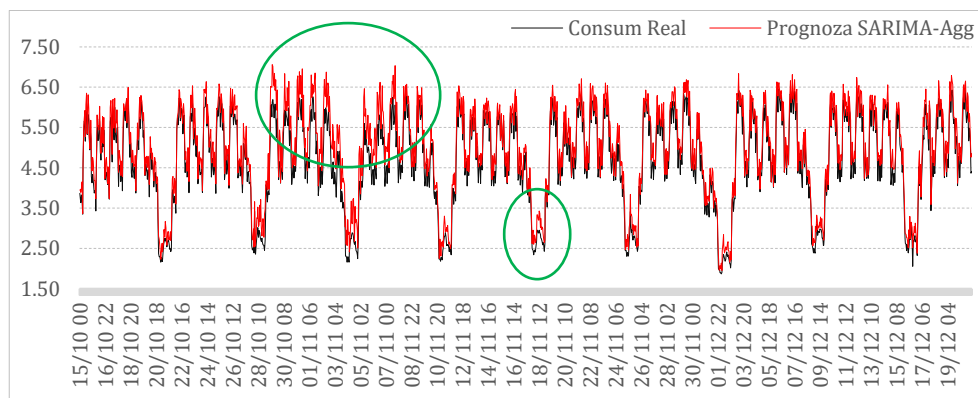


Figura 6.7 Prognoză realizată cu SARIMA (MAPE = 7,56%)

Metoda bazată pe învățare cu fereastră glisantă cu MLP obține un rezultat MAPE sub 6% și reușește să identifice variația consumului în orele de gol de sarcină mai bine decât în orele de vârf. Perioada din jurul sărbătorilor naționale este problematică pentru algoritm, acesta prognozând ziua de luni (02.12.2019) ca fiind o zi de sărbătoare (Figura 6.8).

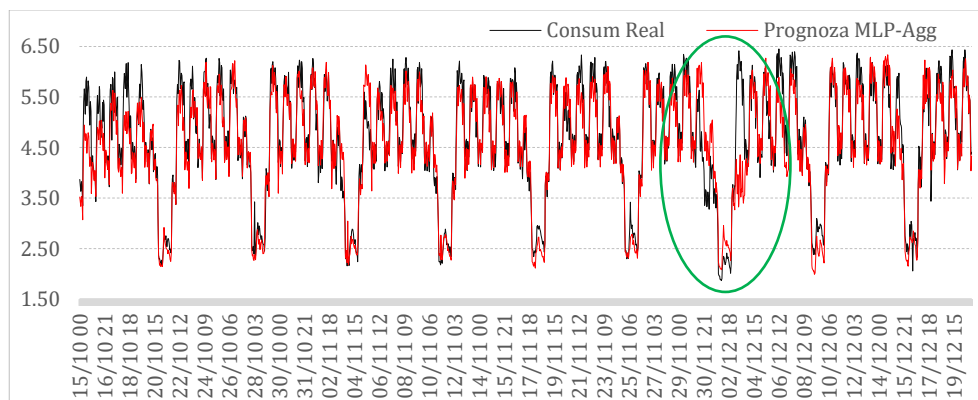


Figura 6.8 Prognoză realizată cu MLP (MAPE = 5,95%)

În Figura 6.9 sunt prezentate rezultatele pentru cea mai simplă rețea neuronală recurentă și se poate observa că se obține un scor mai mic decât cu MLP cu 11,09%. Se identifică aceeași problemă cu perioada de sărbătoare și chiar dacă identifică evoluția consumului în orele de gol

mai bine decât MLP, orele de vârf sunt determinate cu o eroare de peste 6%.

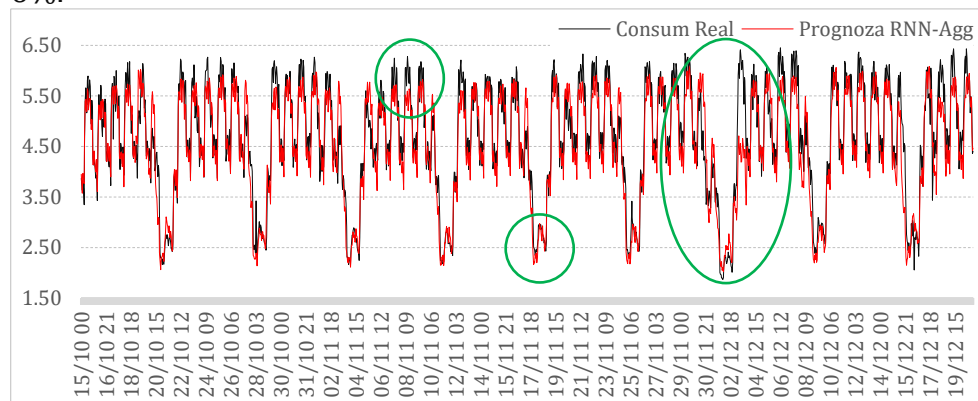


Figura 6.9 Prognoză realizată cu RNN (MAPE = 6,61%)

Metoda LSTM obține un rezultat bun având MAPE 5,67%, aproape de cea mai bună metodă implementată. Chiar dacă estimează bine perioada premergătoare zilei naționale se poate observa în Figura 6.10 că metoda tinde să estimeze spre extremitate valorile de vârf și depășește valorile reale. Algoritmul LSTM este cea mai complexă variantă a rețelelor RNN și este eficient la extragerea tiparelor din caracteristicile de intrare, unde datele de intrare se întind pe secvențe lungi. Având în vedere arhitectura închisă a LSTM-urilor care are această capacitate de a-și regla starea memoriei pe o perioadă mai lungă de timp acesta pune mai mult accent pe variațiile de consum anterioare și nu de trendul actual de variație al consumului.

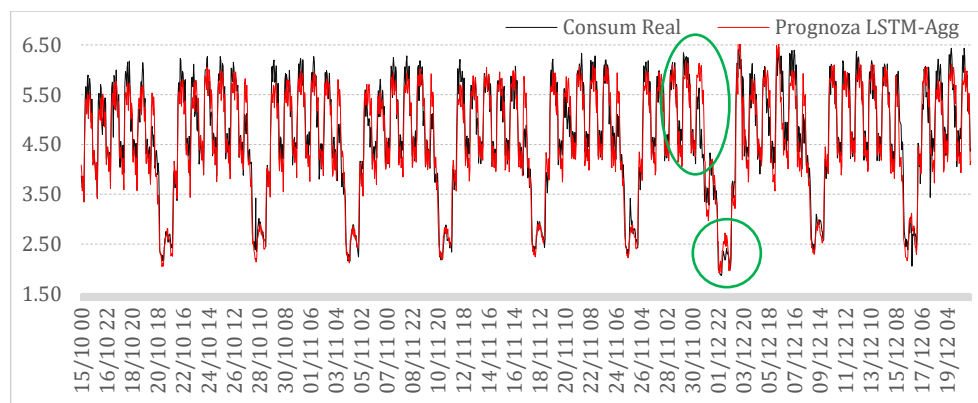


Figura 6.10 Prognoză realizată cu LSTM (MAPE = 5,67%)

Antrenarea secvențială a rețelei LSTM se realizează cu ajutorul procedurii “ecoder-decoder” care facilitează memorearea anumitor secvențe în seriile de timp. Rezultatele sunt prezentate în Figura 6.11 și se poate observa că algoritmul nu identifică corect variațiile în orele de vârf, iar în perioada cu 1 Decembrie prognozează greșit ziua de luni.

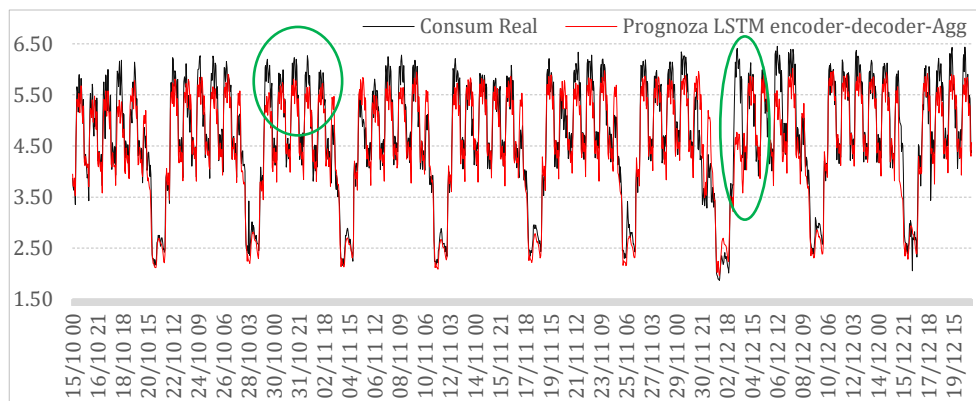


Figura 6.11 Prognoză realizată cu LSTM encoder-decoder (MAPE = 6,23%)

Dintre rețelele neuronale recurente implementate, cele mai bune rezultate se obțin (Figura 6.12) cu algoritmul GRU cu un MAPE de 5,28%. Se poate observa perioada din jurul zilei naționale în care algoritmul identifică revenirea la activitatea normală de muncă, chiar dacă zilele premergătoare estimează un consum mai ridicat. Orele de consum în vârf și gol sunt prognozate corect și metoda obține o eroare medie MAPE superioară celorlalte metode implementate.

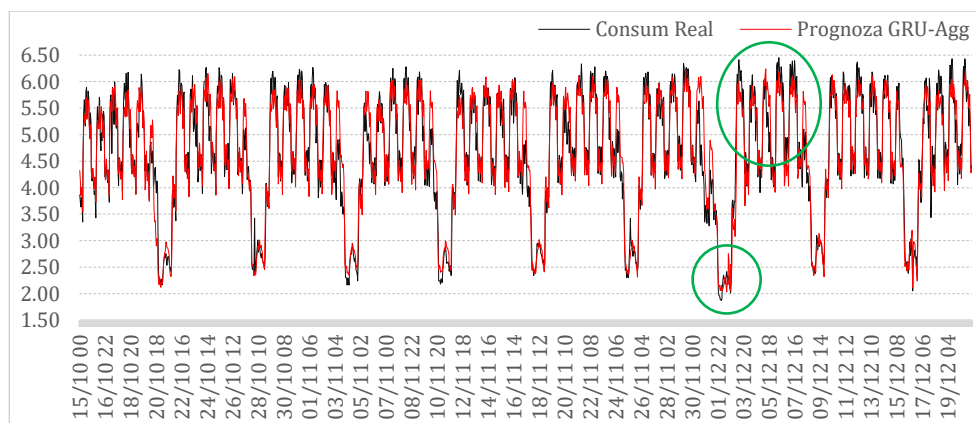


Figura 6.12 Prognoză realizată cu GRU (MAPE = 5,28%)

În Figura 6.13 se prezintă rezultatele metodei CNN-LSTM, care identifică variația orelor de gol, dar nu are rezultate satisfăcătoare pentru orele de vârf și în perioada cu zilele libere de 1 Decembrie.

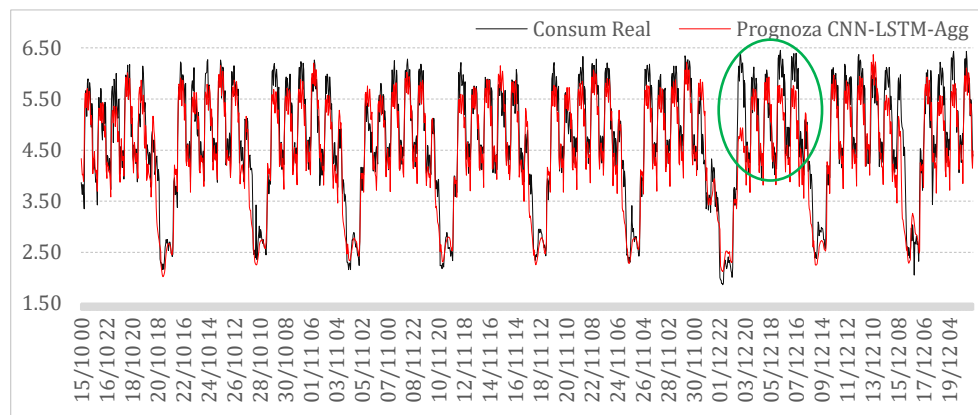


Figura 6.13 Prognoză realizată cu CNN-LSTM (MAPE = 6,97%)

Perioada cea mai nefavorabilă din setul de date de testare este prezentată în detaliu prin analiza următoare pentru a compara performanța algoritmilor. Perioada din data de 27 noiembrie 2019 (miercuri) până la 3 decembrie 2019 (marți) este neobișnuită, deoarece se suprapun două sărbători legale (pe 30 noiembrie și 1 decembrie) cu weekend. Într-o situație ca aceasta, angajații își iau de obicei o zi sau două de vacanță în plus, dar în cazul prezentat problema o reprezintă ziua de luni (02 decembrie 2019) deoarece activitatea consumatorilor revine la valorile normale unei zile lucrătoare. În Figura 6.14 se poate observa că de vineri consumul începe să scadă, sâmbăta este mai mic decât de obicei, iar luni este mai mare decât într-o zi normală de luni, pentru a recupera zilele de neproductivitate. Figura 6.14 prezintă rezultatele metodelor tradiționale. Deoarece acest tip de variație nu a mai avut loc anterior, algoritmi nu reușesc să detecteze vineri, sâmbătă și luni, cu excepția ES și SARIMA. Cea mai mică eroare se obține prin SARIMA cu 8,4% mai bună decât metoda AR pentru această perioadă, care are un MAPE global mai bun pentru perioada de testare. În Figura 6.15 rezultatele metodelor ML sunt prezentate pentru aceeași perioadă menționată anterior. Similar metodelor tradiționale, deoarece vineri este zi lucrătoare, metodele ML prevăd un consum mai mare decât cel real. Totuși, pentru luni, pentru că urmează ziua lucrătoare după zile libere, algoritmi identifică greșit un consum mai mic.

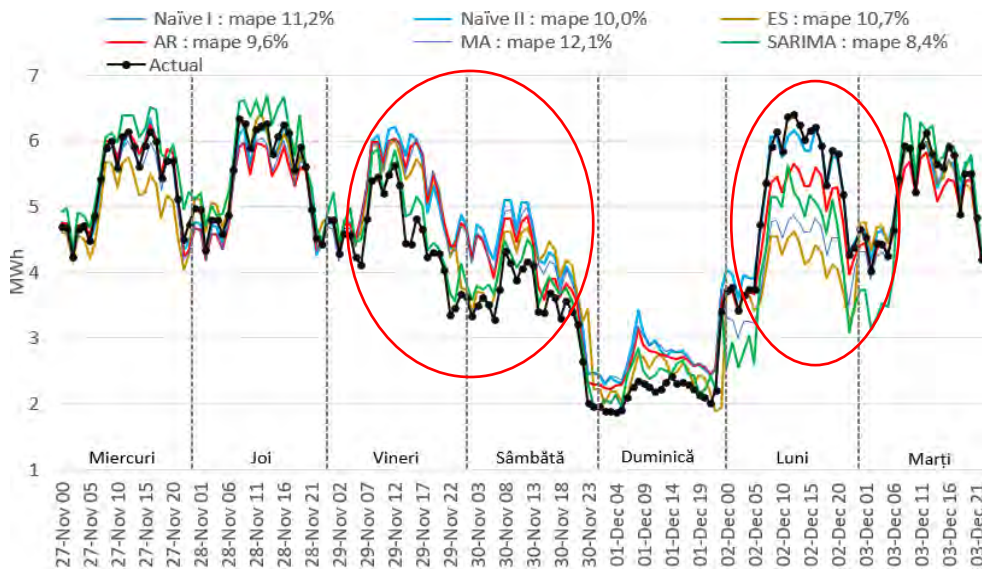


Figura 6.14 Consumului real vs prognozat - metode tradiționale.

Pentru perioada menționată s-au obținut erori mari (12,3% MAPE), iar cu algoritmul MLP, mai proaste decât metodele tradiționale.

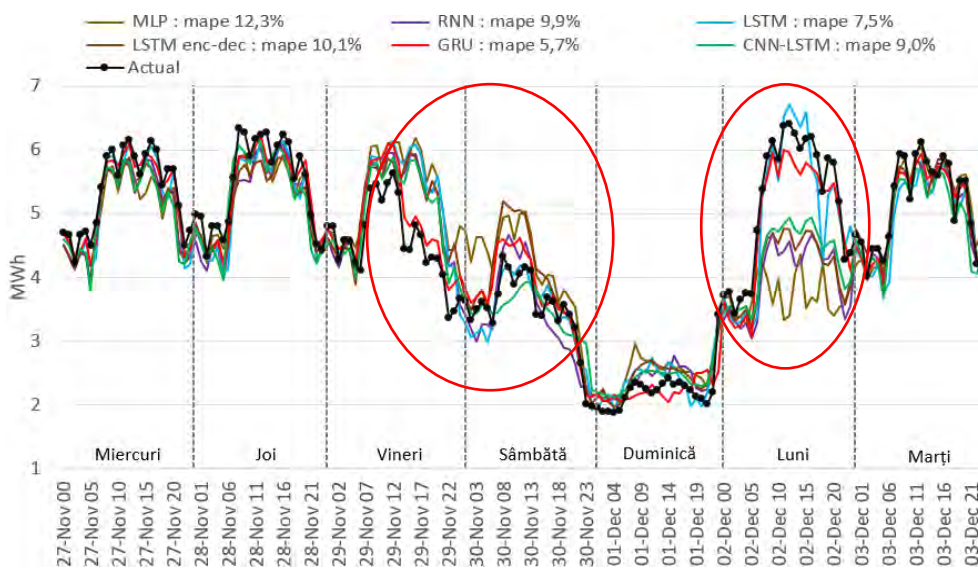


Figura 6.15 Consumului real vs prognozat - ML.

Acest rezultat este similar cu metodele tradiționale pentru “encoder-decoder” LSTM, CNN-LSTM, RNN și MLP. Metodele GRU și LSTM au rezultate bune în această perioadă. Figura 6.16 și Figura 6.17 evidențiază o performanță similară, dar GRU identifică mai bine curba de

sarcină aferentă zilelor de vineri și luni, și astfel depășește rezultatul obținut de metoda LSTM.

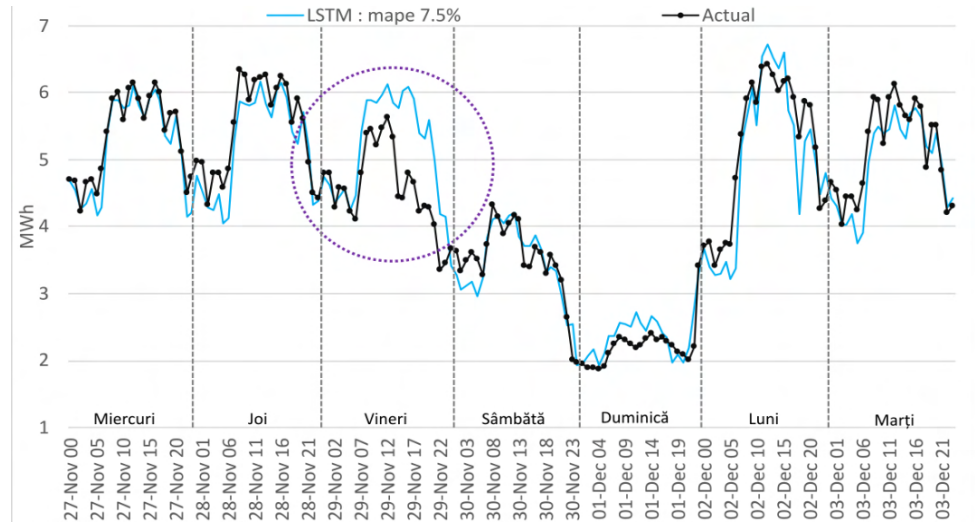


Figura 6.16 Consumului real vs prognozat - LSTM.

LSTM are performanțe mai slabe cu 23,2% decât metoda GRU din cauza sub-adaptării rețelei pe datele de antrenament. Folosirea unui număr similar de epoci pentru a antrena atât GRU cât și LSTM a dus la sub-adaptarea datelor de antrenament pentru LSTM. Mărirea numărului de epoci pentru LSTM a crescut și timpul de învățare dar algoritmul a trecut din sub-adaptarea în supra-adaptarea rețelei fără ca eroarea să se îmbunătățească.

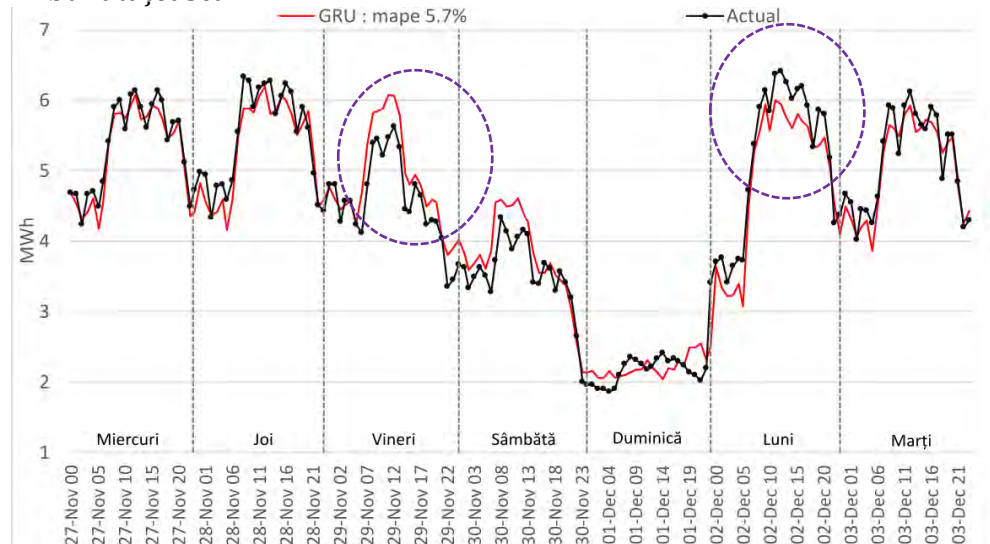


Figura 6.17 Consumului real vs prognozat - GRU.

Rețelele GRU pot fi privite ca o rețea LSTM simplificată, care se antrenează mai rapid și, prin urmare, generalizează mai bine datele de antrenament. În Figura 6.16, metoda LSTM interpretează ziua de vineri ca o zi de lucru normală, iar în Figura 6.17 GRU reușește să minimizeze decalajul dintre valorile reale și cele prognozate.

Metoda GRU a prognozat cu cea mai mică eroare întreaga perioadă de testare și a confirmat cea mai bună performanță pentru perioada nefavorabilă din 27 noiembrie 2019 (miercuri) până pe 3 decembrie 2019 (marți) cu 5,7% MAPE. O altă comparație este prezentată în Figura 6.18 pentru o săptămână normală lucrătoare (din 9 decembrie 2019 până în 15 decembrie 2019) pentru a evidenția diferențele dintre cele mai bune două metode implementate. Prognoza GRU depășește LSTM cu 26,59% cu 3,82% MAPE, în timp ce LSTM a marcat un MAPE de 5,2%. Pentru sâmbătă se poate observa că algoritmi au prognozat o valoare mult mai mică a consumului, situație explicabilă prin influența weekend-ului din analiza anterioară cu sărbătorile legale.

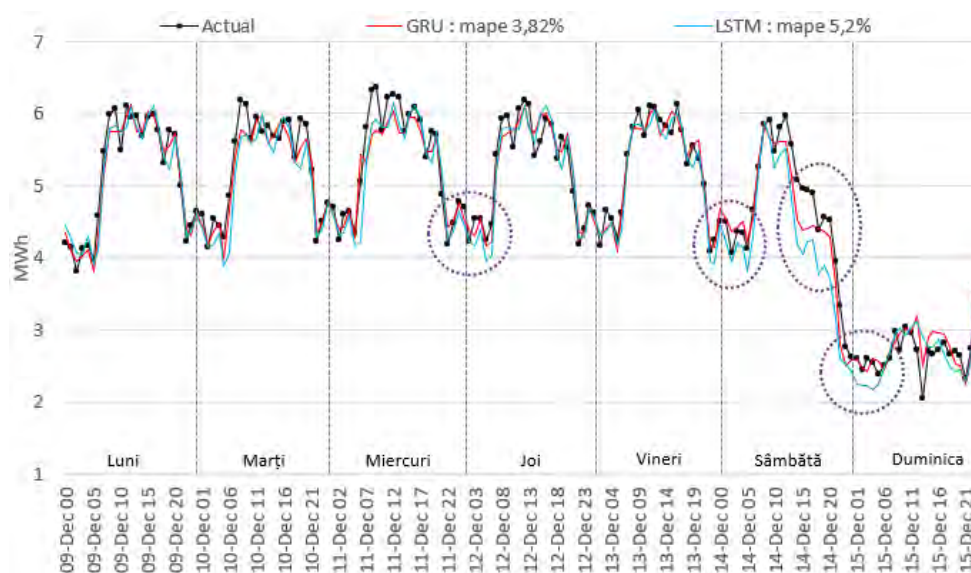


Figura 6.18 Consumului real vs prognozat - GRU și LSTM.

Scopul cercetării efectuate este de a demonstra caracterul practic al învățării automate în prognoza pe termen scurt pentru consumatorii industriali și comerciali agregați. Prognozele sunt validate cu ajutorul indicatorilor statistici pentru erori și prin calcularea impactului pieței pentru ziua următoare și a pieței de echilibrare. Această abordare nouă

în determinarea impactului economic al prognozei în raport cu literatura de specialitate este prezentată în capitolul 8. Pentru a îndeplini acest criteriu practic, orizontul de timp al prognozei trebuie să faciliteze accesul pe platformele de tranzacționare electricitate, respectiv un orizont mai mare de 24 ore. În analizele următoare se prezintă valorile erorilor obținute cu algoritmi ML pentru orizont de timp de 1 oră, 24 ore, 48 ore și 168 ore. În Tabel 6.4 sunt prezentate toate configurațiile considerate pentru obținerea celor mai bune rezultate cu algoritmi ML.

Tabel 6.4 Configurarea rețelelor neuronale în funcție de orizontul de timp prognozat

Neuroni intrare	Straturi ascunse	Număr de neuroni în strat 1	Număr de neuroni în strat 2	Număr de neuroni în strat n	Neuroni ieșire
1	2-10	10-744	10-744	10-744	1
24	2-10	10-744	10-744	10-744	24
48	2-10	10-744	10-744	10-744	96
168	2-10	10-744	10-744	10-744	168

În Tabel 6.5 se prezintă prognozele efectuate pe cele patru orizonturi de timp aplicate pe curba de sarcină agregată. Se evidențiază perioada de antrenare și testare pentru a demonstra performanța pe date necunoscute algoritmilor ML. Dacă măsurile erorilor în perioada de antrenare sunt foarte mici în raport cu cele din perioada de testare, înseamnă că algoritmi ML sunt supra-adaptați pe setul de antrenare. Acest fenomen afectează performanța prognozei, deoarece perioada de testare este scenariul care imită prognoza în practică

Tabel 6.5 Măsurători prognoză pentru algoritmi în funcție de orizontul de prognoză

		Orizontul prognozei : 1 oră						
		LSTM_GRU	LSTM enc dec	LSTM-CNN	LSTM	MLP	simple RNN	GRU
MAPE	învățare	5,82%	6,51%	6,63%	5,53%	6,88%	5,04%	4,52%
		0,3371	0,3768	0,3542	0,3192	0,3895	0,2896	0,2681
		0,2530	0,2765	0,2698	0,2356	0,2996	0,2149	0,1929
MAPE	Test	5,78%	5,97%	6,40%	5,83%	7,01%	6,20%	5,75%
		0,3489	0,3525	0,3860	0,3410	0,4216	0,3753	0,3489
		0,2671	0,2710	0,2867	0,2687	0,3390	0,2878	0,2671

		Orizontul prognozei : 24 ore						
		LSTM_GRU	LSTM enc dec	LSTM-CNN	LSTM	MLP	simple RNN	GRU
MAPE	învățare	4,68%	3,29%	5,72%	3,96%	4,79%	4,14%	3,64%
		0,4206	0,3816	0,4516	0,3969	0,4334	0,3995	0,3895
		0,2030	0,1555	0,2397	0,1767	0,2030	0,1825	0,1662
MAPE	Test	6,13%	6,23%	6,97%	5,67%	6,00%	6,61%	5,28%
		0,2070	0,2060	0,2444	0,1784	0,2334	0,2196	0,1696
		0,2864	0,2822	0,3308	0,2607	0,2754	0,3011	0,2420

Orizontul prognozei : 48 ore								
		LSTM_GRU	LSTM enc dec	LSTM-CNN	LSTM	MLP	simple RNN	GRU
MAPE	Învățare	5,57%	8,27%	13,33%	5,27%	6,47%	6,47%	5,66%
RMSE		0,3385	0,4586	0,6831	0,3180	0,3941	0,3669	0,3605
MAE		0,2050	0,3335	0,5593	0,2018	0,2380	0,2442	0,2104
MAPE	Test	6,10%	7,71%	11,97%	6,46%	6,98%	7,38%	6,48%
RMSE		0,3834	0,4620	0,7237	0,3997	0,3797	0,5000	0,3890
MAE		0,2866	0,3549	0,5850	0,2985	0,2806	0,3467	0,3029

Orizontul prognozei : 168 ore								
		LSTM_GRU	LSTM enc dec	LSTM-CNN	LSTM	MLP	simple RNN	GRU
MAPE	Învățare	5,03%	-	-	5,14%	4,79%	5,57%	4,79%
RMSE		0,2923	-	-	0,3036	0,2822	0,3027	0,2907
MAE		0,1721	-	-	0,1799	0,1692	0,1893	0,1673
MAPE	Test	5,64%	-	-	6,30%	5,93%	6,23%	5,74%
RMSE		0,3328	-	-	0,3972	0,3504	0,3554	0,3398
MAE		0,2517	-	-	0,2860	0,2688	0,2678	0,2625

Pentru prognoza pe 168 de pași nu s-a putut ajunge la o arhitectură care să ofere un rezultat satisfăcător pentru algoritmi LSTM “encoder-decoder” și CNN-LSTM, erorile obținute fiind foarte mari. Explicația acestor erori este volumul foarte mare de date pentru un astfel de orizont de timp care determină o rețea neuronală foarte mare, greu de antrenat. Metoda ferestrei glisante învață tipare în date în funcție de orizontul de timp pentru care se dorește prognoza. În cazul celor 168 de pași, matricea de intrare în rețea va avea $[168 \times 20]$ elemente care trebuie procesate.

Pentru a studia mai în detaliu algoritmi prezentați în această lucrare s-au realizat mai mult simulări pentru a varia orizontul de timp și valida modul de implementare. Prognoza efectuată pentru un orizont de 168 pași, reprezintă o săptămână întreagă. O astfel de prognoză poate fi inclusiv folosită pentru accesarea altor produse pe piața de electricitate pentru achiziție sau vânzare. În contextul actual al pieței de echilibrare, în care decontarea se efectuează la 15 minute și notificările se realizează pentru patru valori într-o oră, un orizont de 168 de pași reprezintă o prognoză pe 42 ore. În studiul următor, având în vedere erorile minime obținute, este prezentat în detaliu algoritmul GRU cu configurația prezentată în Tabel 6.6 pentru variația orară a prognozei pentru întreaga perioadă de testare. Deoarece se pot utiliza o multitudine de structuri pentru RNA s-a creat un cod pentru fiecare algoritm pentru o mai bună

identificare a acestora în lucrare și în anexe. De exemplu în Tabel 6.6 este explicată notația algoritmului GRU|24|3|100|100|48|24 care reprezintă numărul de neuroni în fiecare strat. Acest tip de codificare poate să fie urmărit și în anexe.

Tabel 6.6 Configurația rețelelor GRU folosite în studiu – exemplu codificare

Neuroni intrare	Straturi ascunse	Număr de neuroni în strat 1	Număr de neuroni în strat 2	Număr de neuroni în strat 3	Neuroni ieșire
24	3	100	100	48	24
GRU 24 3 100 100 48 24					

În Figura 6.19 se poate observa măsura zilnică a rezultatelor algoritmului bazat pe GRU. Datorită caracterului stocastic de estimare a parametrilor RNA același algoritm cu aceeași structura nu va furniza același rezultat de fiecare dată. Important este să avem o variație minimă a rezultatelor și astfel să considerăm modelul stabil. Se poate identifica în figurile următoare faptul că în jurul zilei de 1 decembrie algoritmul detectează perioada respectivă ca o perioadă de vacanță prelungită și astfel ziua de luni este prognozată sub valorile normale ale unei zile lucrătoare. Această eroare poate fi observată și în cazul RNA-urilor care sunt antrenate pentru multe epoci și astfel se obține supra-adaptare pe date.

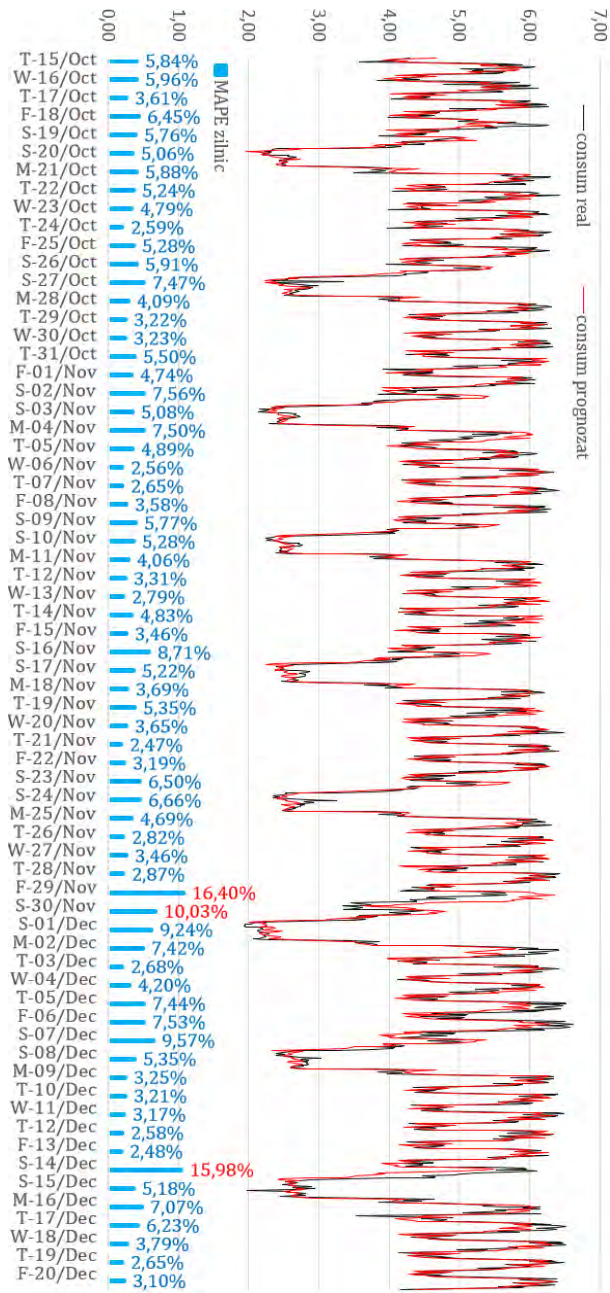


Figura 6.19 Evoluția zilnică a prognozei (GRU|24|3|100|100|48|24)

În Figura 6.20 sunt analizate diferite arhitecturi implementate pentru cea mai bună metodă studiată în această teză (GRU) și se evidențiază atât timpul de antrenare cât și rezultatul MAPE pentru antrenare (A) și testare (T).

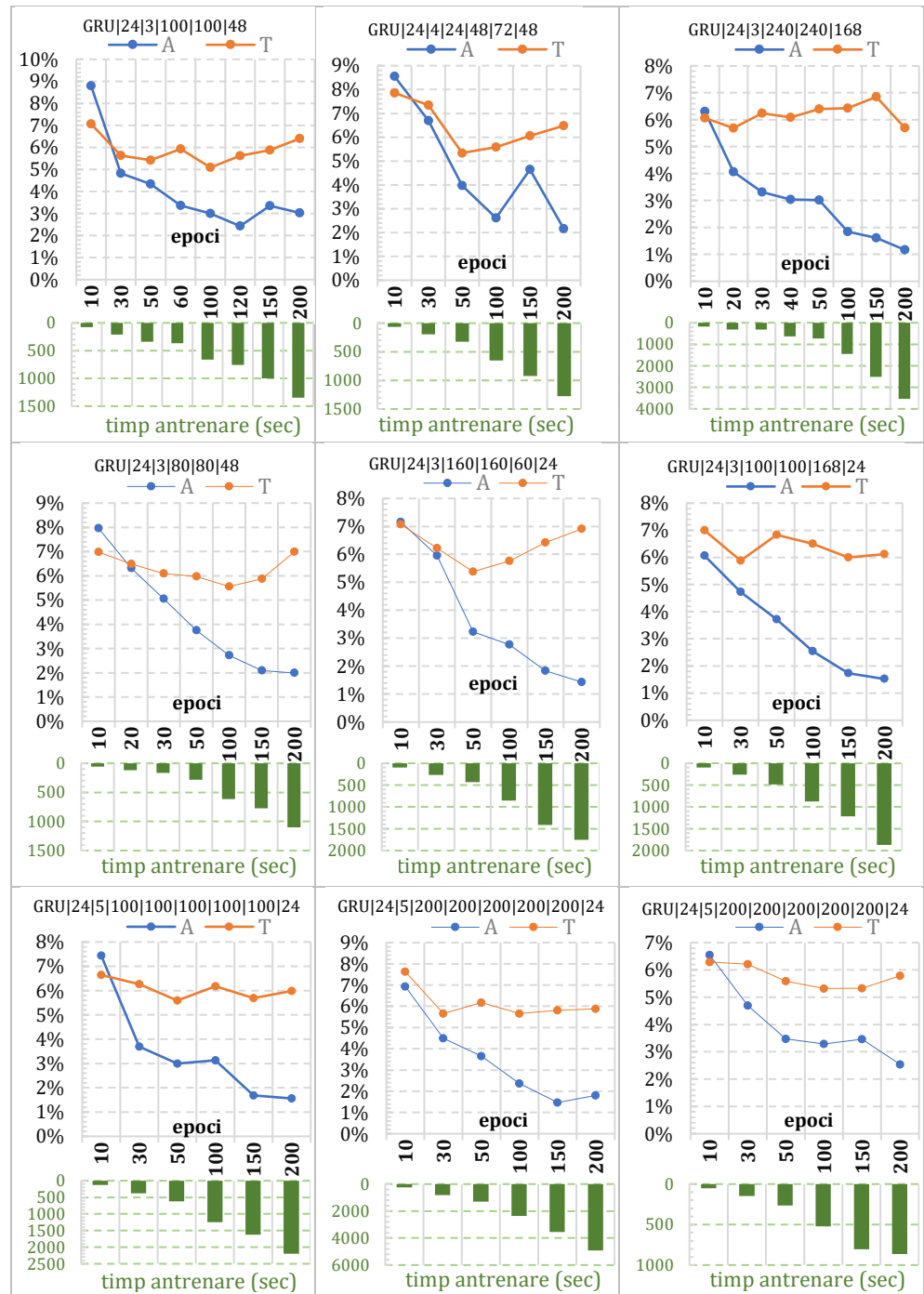


Figura 6.20 Evidențierea diversilor algoritmi GRU antrenați diferit

Procesul de utilizare a algoritmilor DL este stocastic, iar cu valori MAPE diferite punctate prin utilizarea aceleiași arhitecturi, parametrii rețelelor neuronale sunt dificil de reglat fin pentru a se generaliza eficient pe datele de antrenament.

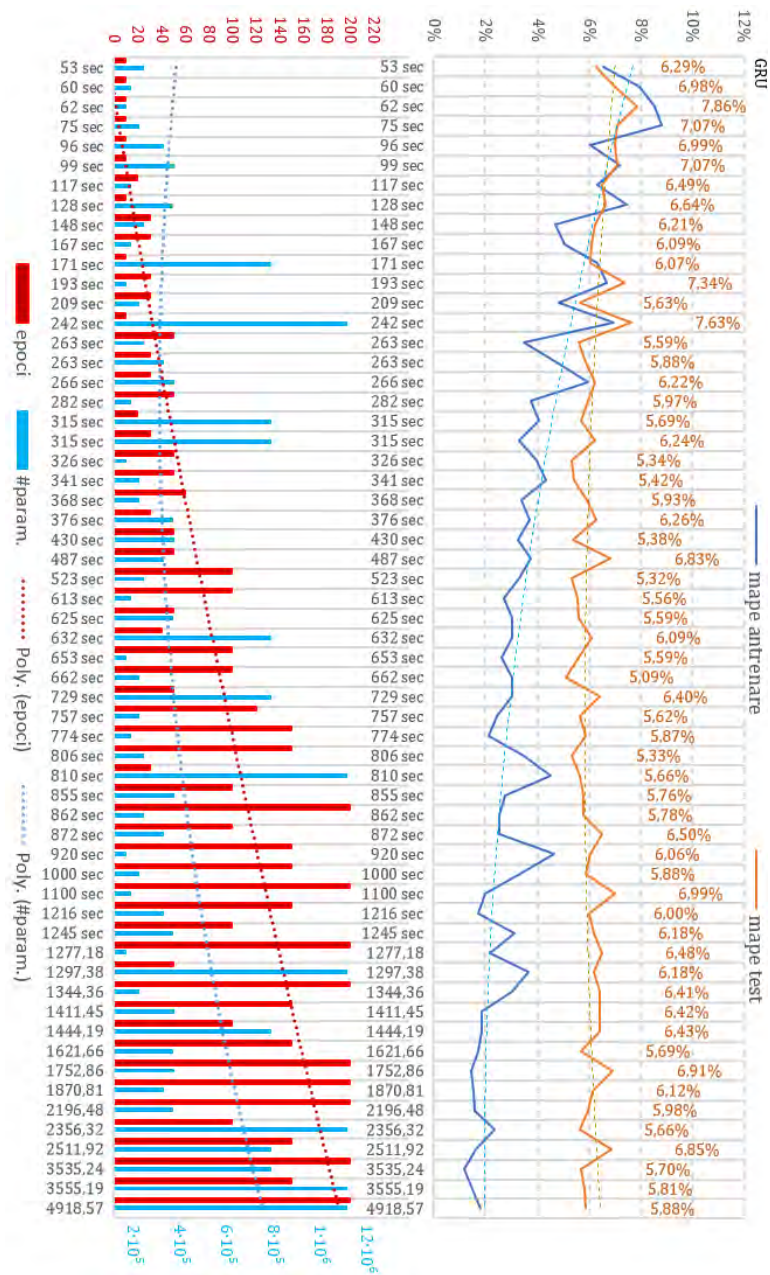


Figura 6.21 Erori obținute cu GRU|24|3|100|100|48|24

Rezultatele indică faptul că utilizarea unei arhitecturi complexe poate duce la erori mai mari din cauza supra-adaptării datelor de antrenament. Cea mai mică eroare de prognoză înregistrată de modelul GRU (MAPE 4,82%) a fost atinsă prin utilizarea unei rețele cu trei straturi ascunse (GRU 24|3|100|100|48|24). Se poate observa că eroarea MAPE de antrenament continuă să scadă, eroarea de test crescând, indicând faptul că rețeaua este supra-adaptată și nu reușește să generalizeze noile date. Această situație trebuie cuantificată pentru a găsi structura care oferă cele mai mici erori.

Pentru identificarea celei mai bune arhitecturi s-a definit un indicator simplu DL_{index} pentru a cuantifica complexitatea algoritmului ML în ecuația (6.3) și pentru a ajuta la compararea rezultatelor algoritmului GRU.

$$DL_{index(i)} = \frac{E_i \times N_{pi}}{E_{max} \times N_{pmax}} \quad 6.3$$

unde: DL_{index} este coeficientul de complexitate, i este numărul de arhitecturi analizate, E este numărul de epoci și N_p este numărul total de parametri utilizați în arhitectura GRU (ponderi și bias). E_{max} este numărul maxim de epoci utilizat în toate simulările, iar N_{pmax} este numărul maxim de parametri din toate simulările. Pentru fiecare simulare, (DL_{index}) este comparat cu MAPE pentru ambele seturi de date (antrenament și test) și timpul de antrenament pentru a ne asigura că DL nu se adaptează prea mult la datele de antrenament.

DL_{index} evidențiază cum este influențată performanța rețelelor neuronale de creșterea numărului de straturi ascunse și de neuroni din fiecare strat. S-a analizat acest experiment și se observă un impact negativ asupra performanței algoritmilor GRU. Devine mai greu din punct de vedere computațional să obținem erori mai mici căci și timpul de antrenament crește în mod nejustificat. Antrenarea unei rețele adânci complexe determină abilitatea fiecărui strat să descrie o caracteristică precisă în relația dintre intrare și ieșire, dar există un punct de inflexiune de la care rețeaua neuronală nu va reuși să se generalizeze pe noi date necesare prognozei și mai târziu pentru achiziția de energie electrică.

În Figura 6.22, este evidențiată evoluția erorii MAPE în comparație cu epocile, timpul de antrenament și complexitatea GRU (DL_{index}). Antrenarea arhitecturilor complexe cu un număr mare de epoci, pe lângă timp și resurse, duce la erori mai mari din cauza supraadaptării.

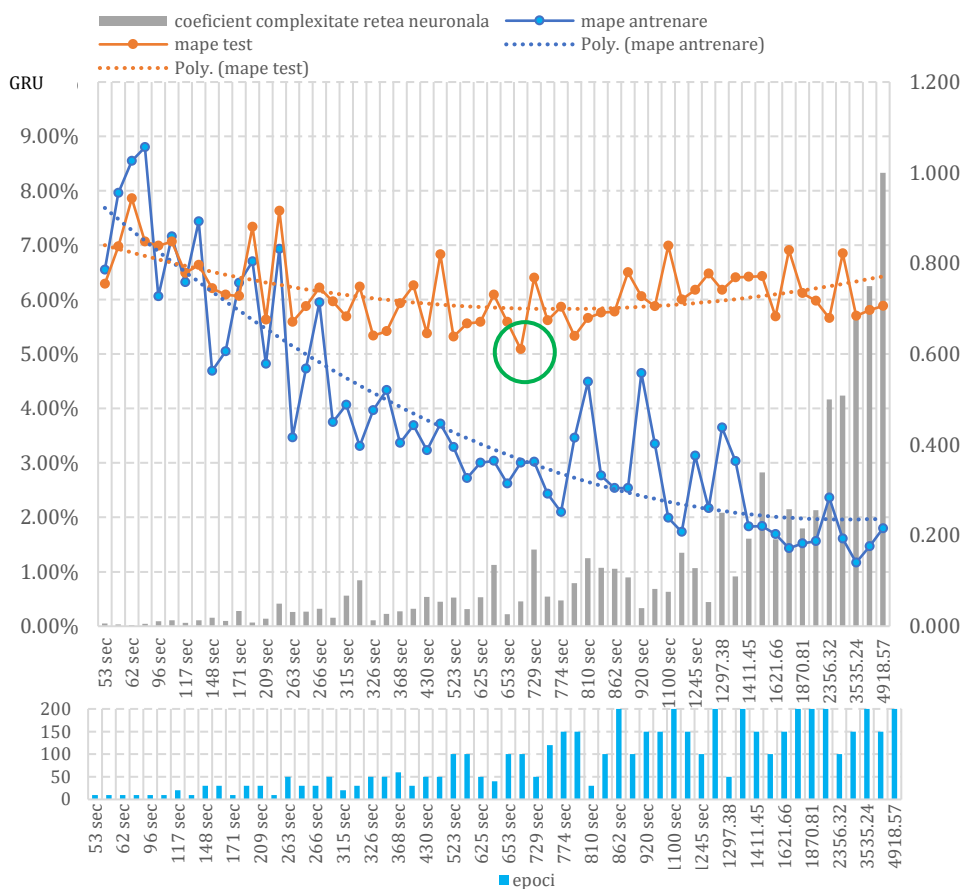


Figura 6.22 Evoluția erorilor în funcție de DL_{index}

În Figura 6.23 se prezintă evoluția indexului de complexitate (DL_{index}) pentru a măsura erorile MAPE atât pentru setul de date de învățare ($MAPE_A$) cât și pentru test ($MAPE_T$). Se poate observa existența unui prag reprezentat de un număr de parametrii (284784) care determină creșterea erorilor atât pentru $MAPE_A$ (învățare) cât și $MAPE_T$ (testare).

Graficele prezentate în Figura 6.23 pentru algoritmul GRU compară evoluția erorii MAPE cu mai multe arhitecturi (numărul de parametrii) antrenate pentru un număr fix de epoci pentru a evidenția corelația între dimensiunea rețelelor și erori.

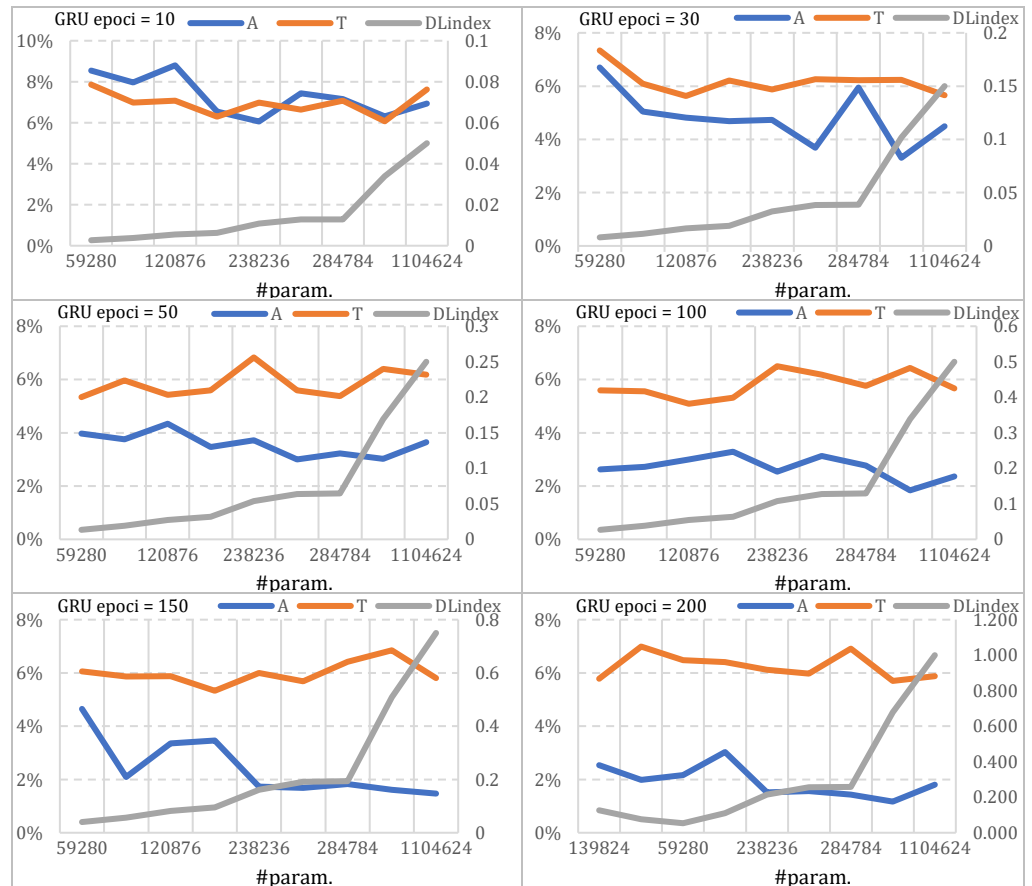


Figura 6.23 Evoluția erorilor în funcție de numărul de parametrii antrenaji

Se pot formula două concluzii foarte importante pentru învățarea automată aplicată prognozei pe termen scurt al consumului agregat de electricitate non-rezidențial:

- i) pe măsură ce crește numărul epocilor crește și diferența dintre curba care reprezintă erorile MAPE în setul de antrenare față de curba pentru rezultatele din setul de test. Folosirea unui număr ridicat de epoci pentru antrenare rezultă în supra-adaptarea rețelei pe datele de învățare și are ca și consecință erori mai mari pentru valori care sunt diferite de setul de antrenare.
- ii) Creșterea numărului de parametrii necesită mai multe epoci pentru antrenare și astfel rețeaua poate să fie sub-adaptată, ceea ce rezultă în erori mari de prognoză. După creșterea numărului de parametrii peste 284.784 eroarea scade pentru setul de învățare doar dacă se ridică numărul de epoci (epoci > 50), dar la

epoci = 200 eroarea crește din nou, după un proces de învățare de două ore. S-a realizat antrenarea unei rețele cu 5.366.964 parametrii pentru 200 epoci (Figura 6.24) care a durat 24 ore și 33 minute. După cum se poate observa eroarea crește, iar la 150 de epoci îmbunătățirea erorii este nesemnificativă față de 100 de epoci și peste valoarea obținută la 50 de epoci.

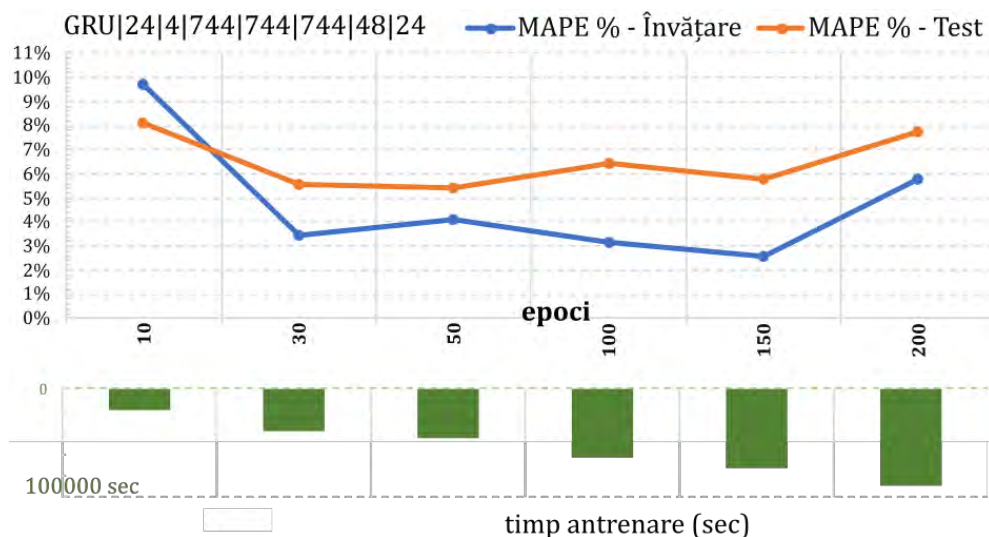


Figura 6.24 Rezultatele arhitecturii GRU cu 5.336.964,00 parametrii

Proгноza de consumului non-rezidențial joacă un rol esențial în costul energiei electrice, mai ales pentru mari consumatori precum cei prezentați în această lucrare. Proгноza consumului este importantă deoarece planificarea furnizării de energie electrică depinde de progноzele de consum. Dezechilibrele mari generate între consumul real și cel progноzat vor crea un risc mai mare pentru toți participanții la piața energiei electrice. Teza își propune o metodologie de progноză industrială bazată pe algoritmi de învățare automată. Rezultatele și analiza indică faptul că acești algoritmi are un grad ridicat de stocasticitate și este necesară reglarea atentă a parametrilor. Provocarea este de a anticipa comportamentul stocastic al marilor consumatori. În lucrare s-a demonstrat că rețelele neuronale adânci pot progноza cu succes sarcina industrială orară. Obstacolul în calea acestei abordări este lipsa contorizării inteligente și a senzorilor pentru progноza consumului în timp real. Fără date relevante, modelele de progноză eficiente și replicabile nu reprezintă o investiție de încredere pentru sectorul privat. Originalitatea tezei constă în algoritmi implementați și cadrul aplicat

pentru prognoza curbelor de sarcină non-rezidențial, analiza celei mai bune arhitecturi și scalabilitatea rețelelor neuronale folosind un indice de complexitate. Studiul a comparat performanța prognozei pentru șapte metode cu învățare automată și a testat diferite combinații pentru variabilele de prognoză și arhitecturi. Rezultatele eșantionului de testare pentru 1.608 de valori orare (15 octombrie-20 decembrie 2019) indică în mod constant că: (i) rețelele neuronale recurente adânci sunt potrivite pentru consumul de energie electrică și (ii) cel mai bun model implementat este GRU. Lucrarea evidențiază influența numărului de straturi ascunse și de neuroni din fiecare strat (parametri totali) asupra performanței algoritmilor ML. Antrenând o rețea adâncă complexă pentru multe epoci, se obține o interpretare exactă între valorile de intrare și ieșire din setul de antrenare dar rețeaua neuronală nu va reuși să identifice variații în datele noi necesare prognozei.

7. Prognoză individuală pentru consumatori

7.1 Introducere

Cea mai bună metodă de prognoză din punct de vedere tehnic al comportamentului consumatorului față de rețeaua de energie electrică este cea individuală. Rețelele electrice inteligente ar putea să faciliteze acest tip de prognoză și să se valorifice în acest fel întregul potențial al prognozelor. Metodele de prognoză pot fi extinse în detaliu la fiecare consumator, dacă monitorizarea consumului este distribuită pe contururi energetice mai mici, acestea pot fi prognozate separat. Din acest studiu rezultă că prognozele agregate oferă rezultate cu valori ale erorilor mai mici decât realizarea individuală a prognozelor și însumarea acestora. În consecință costurile pe piața de electricitate sunt mai mici pentru furnizor. Există un astfel de model de business pe piața românească, respectiv un PRE (parte responsabilă cu echilibrarea) care agreează un procent considerabil din furnizori și consumatori mari. Într-adevăr costurile de echilibrare pentru clienții acestui PRE sunt mai mici față de piața de echilibrare și decât dacă ar face prognoze zilnice pe fiecare consumator. Convenabil financiar este apartenența la un PRE mare, aspect care are drept consecință directă lipsa interesului pentru disciplinarea consumului de energie electrică. Din punct de vedere tehnic pentru rețeaua electrică nu există nici un beneficiu pentru o astfel de abordare în care se echilibrează furnizorii într-un PRE, dimpotrivă

consecința este un dezinteres din partea furnizorilor și a consumatorilor pentru echilibrarea curbelor de sarcină. Această lucrare utilizează algoritmi cu învățare automată pentru variațiile orare ale predicției consumului de energie electrică folosind vremea, tipul de zi, ziua săptămânii, variabile autoregresive ca intrare în setul de date de antrenament și testare. Datele istorice utilizate ca intrare în rețele sunt selectate pe baza rezultatelor observate. Cele mai bune rezultate obținute folosesc ultimele două săptămâni de consum orar ca intrare în rețelele neuronale. Perioade mai scurte de timp utilizate în procesul de învățare au crescut măsura erorilor MAPE. Acest aspect înseamnă tipare zilnice puternice care sunt repetitive săptămânal. Pentru metoda autoregresivă (AR), decalajul autoregresiv a fost selectat pe baza analizei valorii statistice *P-value* aplicate pentru două săptămâni. Metoda AR este o implementare de regresie a datelor din seria temporală pentru a prezice valori viitoare pe baza corelațiilor trecute. Pentru a prognoza ziua ($d+1$) pentru fiecare oră ($h_{1..24}$) se consideră ultimele două săptămâni ($d-1, d-2 \dots d-14$) cu fiecare oră ($h_{1..24}$).

7.2. Metodă și implementare

Pentru prognozele individuale s-a ținut cont de caracteristicile și fluxul tehnologic al fiecărui consumator. Astfel s-a stabilit care sunt factorii exogeni care trebuie luați în calcul pentru realizarea prognozelor. Cea mai puternică corelație identificată între consum și factori exogeni este între consumul unui supermarket și temperatură (Figura 7.1 și Figura 7.2). Se pot observa în figurile următoare variațiile ciclice și sezoniere, datorate în principal de nevoia de agent termic necesar menținerii alimentelor proaspete și confortul clienților.

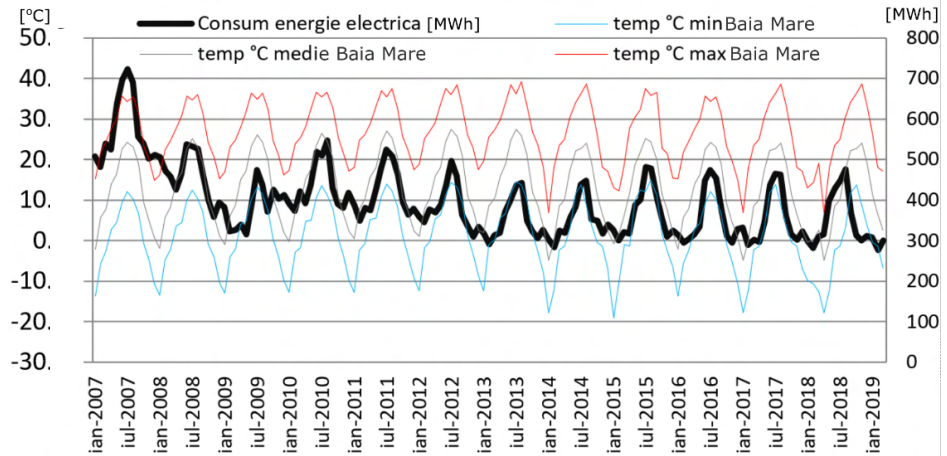


Figura 7.1 Influența temperaturii în consumul de electricitate

— 2007 — 2008 — 2009 — 2010 — 2011
 — 2012 — 2013 — 2014 — 2015 — 2016

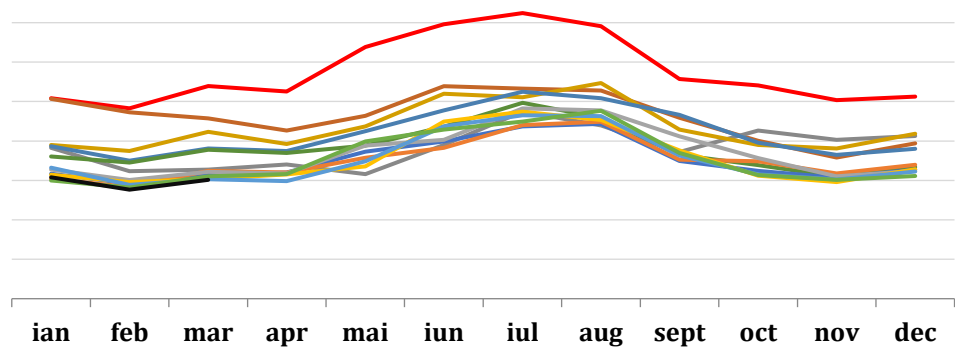


Figura 7.2 Influența temperaturii în consumul lunar al supermarket-ului

Consumatorii industriali, care reprezintă cel mai important consum în grupul de consumatori analizați, își desfășoară activitatea în domeniul prelucrării lemnului și au un flux tehnologic compus din toate procesele necesare prelucrării lemnului pornind de la lemn brut până la produsele finite. Necesarul de energie al consumatorilor sunt energia electrică și energia termică obținută în centrale termice care funcționează cu resturi de lemn. Producția de căldură și apă caldă se bazează pe arderea lemnului rămas din procesele tehnologice. Încălzirea în perioada de iarnă pentru clădirea de birouri și instalațiile de producție din fabrică se realizează cu încălzitoare electrice care influențează consumul în perioada de iarnă împreună cu sistemele de iluminat (programul de lucru este în trei schimburi). Consumul mare de energie electrică este realizat de către hale mari de depozitare și uscare lemn,

utilizate pentru prepararea termică a lemnului brut. Se constată corelația între sarcina electrică și temperatura exterioară, punctul de rouă și umiditate, motiv pentru care acești factori exogeni au fost folosiți în prognoză. Tiparele de consum în zilele lucrătoare/nelucrătoare nu sunt aceleași, deoarece planificarea fabricilor depinde în mare măsură de cota de producție. Un test statistic Dickey-Fuller [111] realizat pentru seriile de timp orare pentru întregul istoric indică caracterul staționar a seriilor de timp. Dependente liniare relevante între variabilele exogene și consum au putut fi stabilite, iar algoritmi cu învățarea automată au devenit o opțiune de explorat pentru dependente neliniare. Dintre toți algoritmi implementați în acest capitol, variantele rețelelor neuronale recurente (LSTM, GRU, GRU-LSTM) oferă cele mai bune rezultate și dintre acestea algoritmul GRU oferă cel mai bun rezultat pentru prognozele individuale.

În tabelul 7.1 se prezintă pentru fiecare algoritm implementat parametrii luați în calcul pentru realizarea prognozelor pe 24 de ore pentru curbele individuale de consum.

Tabel 7.1 Parametrii considerați pentru evaluarea algoritmilor de prognoză individuală

Metodă	Parametrii considerați
AR	Predicția autoregresivă pentru fiecare oră se bazează pe aceeași oră din ultimele 14 zile. Coeficienții sunt prezentați în tabelul 2.
MLP	Perceptron cu mai multe straturi. Matricea de intrare [24,11]. Variabila de intrare: Ultimele 14 zile, Ziua săptămânii, Zi și ore lucrătoare/nelucrătoare, zile speciale, temperatură. 2×straturi ascunse (300, 200). Strat de ieșire: Dens; Activare: Sigmoid; Optimizator: Adam; Pierdere: MSE; Epoci: 100.
Simple RNN	Rețea neuronală recurentă. Matrice de intrare[24,11]. Variabilă de intrare: Ultimele 14 zile, Ziua săptămânii, Zi și ore lucrătoare/nelucrătoare, zile speciale, temperatură, umiditate, punct de rouă. 3×straturi ascunse (100, 100,96). Strat de ieșire: Dens; Activare: Tanh, Sigmoid; Optimizer: Adam; Pierdere: MSE; Epoci: 100.
LSTM	Memoria pe termen lung. Matrice de intrare[24,11]. Variabila de intrare: Ultimele 14 zile, Ziua săptămânii, Zi și ore lucrătoare/ nelucrătoare, zile speciale, temperatură, umiditate, punct de rouă. 3×straturi ascunse (100, 100,168). Strat de ieșire: Dens; Activare: Tanh, Sigmoid; Optimizer: Adam; Pierdere: eroare pătratică medie; Epoci: 100.
LSTM encoder decoder	Encoder-decoder LSTM Matrice de intrare[24,11]. Variabilă de intrare: Ultimele 14 zile, Ziua săptămânii, Zi și ore lucrătoare/ nelucrătoare, zile speciale, temperatură, umiditate, punct de rouă. 3 × straturi ascunse (100, 100, 100), 1 × Repeat Vector, 1 × Strat distribuit în timp (96). Activare: Tanh, Sigmoid; Optimizator: Adam; Pierdere: eroare pătratică medie; Epoci: 100.

GRU	Unitate recurentă închisă. Matrice de intrare[24,11]. Variabila de intrare: Ultimele 14 zile, Ziua săptămânii, Zi și ore lucrătoare/ nelucrătoare, zile speciale, temperatură, umiditate, punct de rouă. 3×straturi ascunse (100, 100, 48). Strat de ieșire: Dens; Activare: Tanh, Sigmoid; Optimizer: Adam; Pierdere: eroare pătratică medie; Epoci: 100.
GRU + LSTM	GRU+LSTM Combinație de straturi LSTM și GRU. Matrice de intrare[24,11]. Variabila de intrare: Ultimele 14 zile, AR(9), Ziua săptămânii, Zi și ore lucrătoare/ nelucrătoare, zile speciale, temperatură, umiditate, punct de rouă. 3×straturi ascunse (100, 100, 48). Ieșire: Dens; Activare: Tanh, Sigmoid; Optimizer: Adam; Pierdere: MSE; Epoci: 100.

7.3. Rezultate

Analiza rezultatelor este prezentată în Tabel 7.2 și arată că erorile sunt mai mari pentru prognoza individuală față de cea agregată. Cele mai mici erori cu MAPE de 5% se obțin prin aplicarea metodelor ML pentru consumul agregat și individual în perioada de testare. Erorile de prognoză pentru curba de sarcină a supermarket-urilor sunt cele mai mici cu măsura MAPE de 4,20%. Motivul pentru această eroare redusă este caracteristica liniară a seriei de timp și corelația cu temperatura. Un supermarket folosește electricitate pentru echipamente care funcționează în fiecare zi în același mod, singura excepție fiind câteva zile pe an pentru sărbători și zile speciale. Cel mai mare consumator pentru un supermarket sunt instalațiile frigorifice, care depind de temperatură și numărul de clienți din zona de vânzare.

Tabel 7.2 Rezultate prognoză cu metode tradiționale – consum individual

Consum		metric	Naive I	Naive II	ES	AR	MA	SARIMA
PF	învăț.	MAPE	20,78%	28,55%	14,97%	19,07%	19,30%	9,05%
		RMSE	0,4190	0,5668	0,3379	0,3577	0,1656	0,2155
		MAE	0,2253	0,3749	0,2158	0,2248	0,2252	0,1252
	test	MAPE	25,16%	24,85%	19,76%	19,12%	29,32%	14,47%
		RMSE	0,1780	0,2472	0,0908	0,1859	0,0244	0,2065
		MAE	0,2881	0,3211	0,2615	0,2283	0,3231	0,1217
FF	învăț.	MAPE	14,52%	40,26%	14,82%	14,53%	14,51%	11,84%
		RMSE	0,4327	0,8891	0,2429	0,3519	0,4051	0,2386
		MAE	0,1984	0,5490	0,2046	0,1988	0,2365	0,2047
	test	MAPE	5,00%	32,18%	6,42%	5,18%	6,91%	10,56%
		RMSE	0,0207	0,7136	0,0223	0,0202	0,0617	0,0876
		MAE	0,1066	0,5158	0,1174	0,1098	0,1592	0,2508
SM	învăț.	MAPE	6,78%	5,56%	8,01%	4,76%	5,67%	15,68%
		RMSE	0,0319	0,0266	0,0408	0,0223	0,0267	0,0699
		MAE	0,0250	0,0201	0,0315	0,0173	0,0209	0,0560

	test	MAPE	5,54%	4,99%	10,52%	4,22%	5,27%	18,96%
		RMSE	0,0005	0,0004	0,0027	0,0003	0,0005	0,0046
		MAE	0,0183	0,0164	0,0394	0,0139	0,0173	0,0580
FPF	învăț.	MAPE	27,09%	39,57%	17,38%	25,43%	30,69%	17,52%
		RMSE	0,0706	0,1006	0,0904	0,0583	0,0676	0,0589
		MAE	0,0467	0,0620	0,0634	0,0413	0,0477	0,0475
	test	MAPE	31,07%	44,85%	14,05%	29,78%	36,60%	23,32%
		RMSE	0,0046	0,0087	0,0063	0,0032	0,0043	0,0026
		MAE	0,0447	0,0583	0,0529	0,0396	0,0462	0,0415
AS	învăț.	MAPE	8,75%	15,78%	19,48%	7,70%	7,63%	15,46%
		RMSE	0,0085	0,0150	0,0138	0,0070	0,0072	0,0088
		MAE	0,0044	0,0076	0,0091	0,0038	0,0038	0,0068
	test	MAPE	8,55%	15,13%	17,92%	8,05%	7,87%	17,45%
		RMSE	0,0001	0,0002	0,0002	0,0001	0,0001	0,0001
		MAE	0,0043	0,0073	0,0075	0,0040	0,0040	0,0073

În cazul consumatorilor industriali mari, respectiv fabrica de parchet (GRU MAPE - 13,82%) și fabrica de mobilă (GRU MAPE - 4,71%), care influențează agregarea sarcinii, a fost dificil să se identifice variabila exogenă relevantă care poate influența consumul. Eroarea mare pentru PF se datorează faptului că fabrica are echipamente vechi și apar defecțiuni frecvente. Două săptămâni pe an, fabrica are și o revizie generală atunci când comportamentul de consum este foarte volatil. Fluxul de producție se bazează în principal pe motoare electrice care alimentează utilaje care prelucrează lemnul brut până la produsul final. Ambele fabrici au depozite mari ventilate pentru uscarea lemnului. Nu s-a putut stabili nicio relație între temperatura de rouă exterioară sau umiditate. Chiar și zilele lucrătoare/ nelucrătoare nu sunt aceleași, deoarece planificarea fabricii depinde de cota de producție. Dependența de temperatură este scăzută deoarece energia termică este produsă, prin arderea gazelor naturale sau a resturilor de lemn. Pentru consumatorii comerciali sau rezidențiali, metodele de prognoză pot fi îmbunătățite dacă se utilizează variabile exogene, cum ar fi temperatura, umiditatea, radiația solară sau precipitațiile. În articolul [19] autorii prezintă impactul variabilelor exogene menționate în prognoza încărcării. În contrast cu cele menționate de articolul anterior, în articolul [21], autorii evidențiază că utilizarea temperaturii ca variabilă exogenă nu îmbunătățește acuratețea prognozei zilei următoare.

Metoda AR are cele mai mici erori (Agg - 6,76% MAPE) în comparație cu alte metode tradiționale pentru curbele de sarcină agregate și individuale (SM - 4,20% MAPE). Pentru a compara fiabilitatea abordărilor pentru un furnizor de energie, am analizat prognoza curbei

de sarcină agregată în contrast cu însumarea previziunilor individuale de sarcină. Prognoza curbei de sarcină agregată oferă erori mai mici, în special pentru metodele ML. În niciunul dintre cazurile prezentate, prognozele individuale nu au obținut o eroare mai mică decât abordarea agregată. Rezultatul agregat GRU a îmbunătățit rezultatul în comparație cu prognoza individuală cu 8,81%.

Tabel 7.3 Rezultate prognoză cu metode ML – consum individual

Tip		Indice eroare	MLP	RF	RNN	LSTM	LSTMed	GRU	CNN LSTM
FP	învăț,	MAPE	8,85%	3,57%	8,85%	4,52%	5,25%	4,61%	11,00%
		RMSE	0,2129	0,0760	0,1544	0,1599	0,1495	0,2489	0,0000
		MAE	0,1172	0,04	0,1214	0,0764	0,0861	0,0748	0,1499
	test	MAPE	20,91%	14,83%	15,12%	15,98%	15,44%	13,82%	14,84%
		RMSE	0,1112	0,3345	0,1387	0,1358	0,0887	0,0880	0,0000
		MAE	0,2684	0,2249	0,2136	0,2157	0,2132	0,1817	0,1965
FF	învăț,	MAPE	7,19%	2,50%	7,30%	4,36%	4,01%	4,06%	3,19%
		RMSE	0,2525	0,0701	0,2240	0,1953	0,1950	0,1894	0,1600
		MAE	0,1212	0,04	0,1259	0,0866	0,0776	0,0771	0,1675
	test	MAPE	4,94%	4,96%	5,63%	5,15%	6,28%	4,71%	10,31%
		RMSE	0,0341	0,1419	0,0478	0,0440	0,0481	0,0100	0,0853
		MAE	0,1055	0,1392	0,1510	0,1403	0,1432	0,1276	0,2014
SM	învăț,	MAPE	4,49%	1,64%	7,35%	4,58%	4,25%	4,97%	5,73%
		RMSE	0,0312	0,0076	0,0432	0,0315	0,0303	0,0456	0,0345
		MAE	0,0175	0,01	0,0310	0,0179	0,0166	0,0349	0,0213
	test	MAPE	4,29%	4,32%	7,13%	4,33%	4,20%	4,37%	5,34%
		RMSE	0,0021	0,0179	0,0027	0,0021	0,0021	0,0029	0,0022
		MAE	0,0184	0,0142	0,0297	0,0182	0,0180	0,0344	0,0218
FPF	învăț,	MAPE	12,17%	4,21%	11,00%	6,51%	5,52%	8,17%	8,98%
		RMSE	0,0295	0,0109	0,0375	0,0192	0,0182	0,0214	0,0249
		MAE	0,0202	0,01	0,0279	0,0114	0,0101	0,0138	0,0163
	test	MAPE	16,58%	12,41%	17,21%	12,89%	13,43%	14,94%	14,84%
		RMSE	0,0018	0,0307	0,0021	0,0014	0,0015	0,0016	0,0015
		MAE	0,0278	0,0200	0,0328	0,0218	0,0217	0,0242	0,0243
AS	învăț,	MAPE	6,51%	0,25%	7,02%	6,46%	4,35%	8,84%	7,02%
		RMSE	0,0061	0,0007	0,0099	0,0054	0,0047	0,0065	0,0066
		MAE	0,0033	0,00	0,0075	0,0031	0,0023	0,0041	0,0037
	test	MAPE	8,91%	7,58%	8,98%	9,71%	7,73%	7,69%	9,36%
		RMSE	0,0001	0,0061	0,0001	0,0001	0,0001	0,0001	0,0001
		MAE	0,0046	0,0017	0,0085	0,0047	0,0040	0,0055	0,0048

Pentru aprofundarea analizei metodelor de prognoză individuală s-a analizat curba de sarcină a fabricii de mobilier (FF) și s-au implementat mai multe metode ML pentru a se obține cele mai bune rezultate. De asemenea, s-a mai implementat o rețea neuronală

recurentă hibridă cu două straturi ascunse: un strat GRU și un strat LSTM. Prin combinarea rețelelor GRU cu LSTM se încearcă complementarea memoriei LSTM pe termen lung cu abilitatea GRU de a generaliza pe datele de învățare. În Tabel 7.4 se prezintă măsurile erorilor obținute, de unde se poate observa faptul că metoda GRU are cele mai mici erori.

Tabel 7.4 Erori de prognoză pentru setul de date de testare pentru consumator FF

	AR(9)	LSTM enc-dec	LSTM- GRU	LSTM	MLP	Simple RNN	GRU
MAPE	5,53%	6,28%	5,14%	5,43%	5,71%	6,63	4,82%
RMSE	0,146	0,193	0,138	0,141	0,165	0,181	0,131
MAE	0,112	0,145	0,1001	0,104	0,129	0,140	0,0998

În Figura 7.3 este prezentată o săptămână normală de lucru (de luni până vineri) pentru a pune în evidență îmbunătățirea obținută prin utilizarea metodologiei propuse. Se poate observa că utilizarea componentei AR ajută atunci când comportamentul consumatorului este repetitiv și îmbunătățește prognoza.

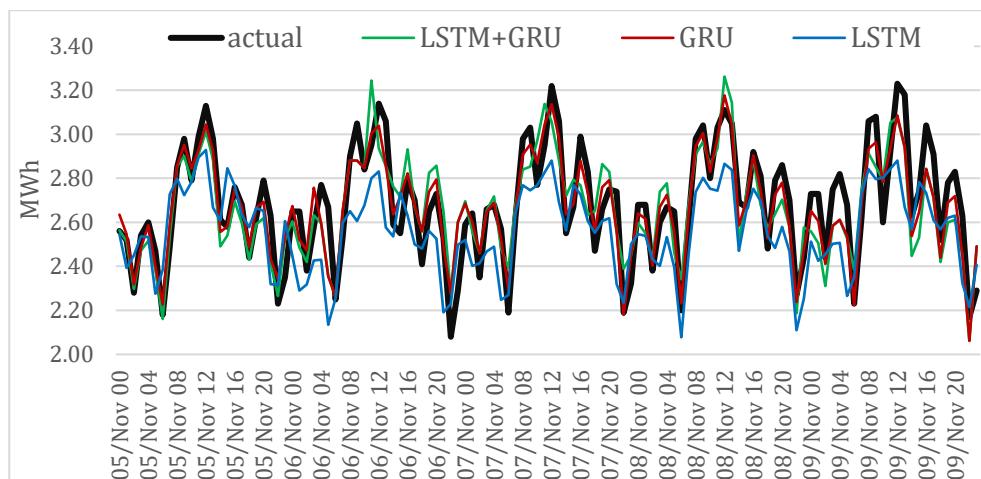


Figura 7.3 Consum real vs prognozat - GRU, GRU+LSTM și AR.

Algoritmii ML au înregistrat rezultate satisfăcătoare pentru curba de sarcină individuală a consumatorului FF, dintre care GRU se evidențiază ca fiind superior. În Figura 7.4 pot fi observate variații mari în orele de vârf de la o zi lucrătoare la alta, raportat la orele de gol, ceea ce determină erori mai mari pentru prognoză. Rețeaua recurentă simplă (RNN) a obținut cel mai slab punctaj de 6,63% MAPE, chiar mai rău decât modelul autoregresiv AR(9). Arhitectura LSTM “encoder-decoder” a obținut rezultate mai slabe decât LSTM cu 14,89%, deoarece arhitectura

LSTM “encoder-decoder” necesită mai multe ponderi antrenabile pentru fiecare dintre pașii de timp ai encoder-ului. Consecința acestei situații este un număr mare de parametri datorat modului în care encoder-ul procesează datele de intrare pentru o serie de timp lungă, chiar mai mulți parametri decât LSTM. O arhitectură neuronală complexă are nevoie de un timp mai lung pentru antrenament, iar problema de sub-adaptare determina metoda LSTM “encoder-decoder” să funcționeze mai rău decât LSTM. Deoarece s-a implementat o structură similară pentru toate rețelele RNN (număr de straturi ascunse, neuroni în straturile ascunse, funcții de activare, algoritmi de învățare), putem concluziona că arhitectura LSTM “encoder-decoder” ar putea îmbunătăți rezultatele prognozirii dacă această rețea complexă ar fi antrenată pentru o perioadă mai lungă cu o resursă hardware superioară celei disponibile.

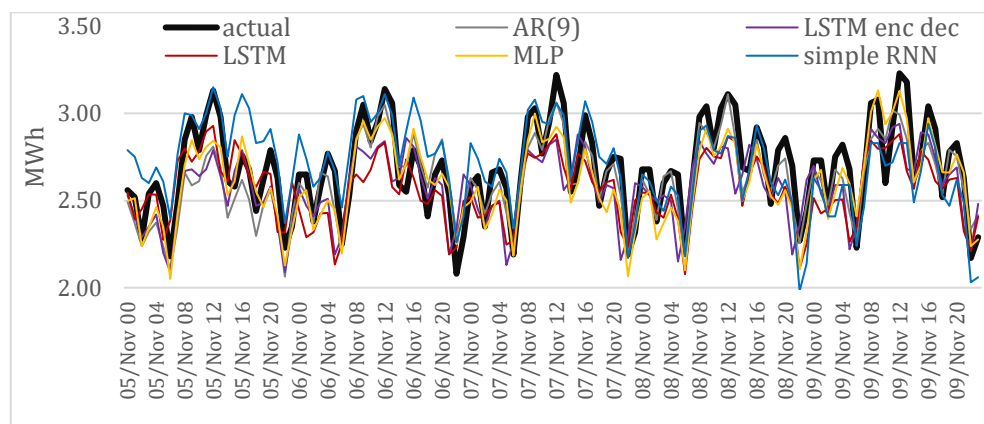


Figura 7.4 Consum real vs prognozat - AR, LSTM, MLP și RNN.

În articolul [46], autorii compară LSTM cu GRU pe seturi de date text și concluzionează că, prin cercetare empirică, avantajul GRU este relevant în scenariul seturilor de date mici. În alte scenarii, în comparație cu LSTM, pierderea de performanță a GRU scade. GRU poate “uita” și alege memoria cu o singură poartă și mai puțini parametri, în timp ce LSTM trebuie să folosească mai multe porți și mai mulți parametri pentru a finaliza aceeași sarcină. Pentru scenariul nostru, cu prognoza sarcinii industriale pe termen scurt în Figura 7.5, rețeaua GRU oferă rezultate mai bune decât LSTM și rețelele combinate GRU și LSTM, rezultate care se pot observa și în Figura 7.6. Deoarece în etapa de învățare a fost utilizată aceeași arhitectură pentru rețelele RNN, LSTM este insuficient antrenat pe setul de date de antrenare pentru a putea să generalizeze pe datele de test. Construirea unei rețele mai mari și creșterea numărului de epoci nu

a îmbunătățit erorile din cauza supra-adaptării rețelei pe datele de antrenare.

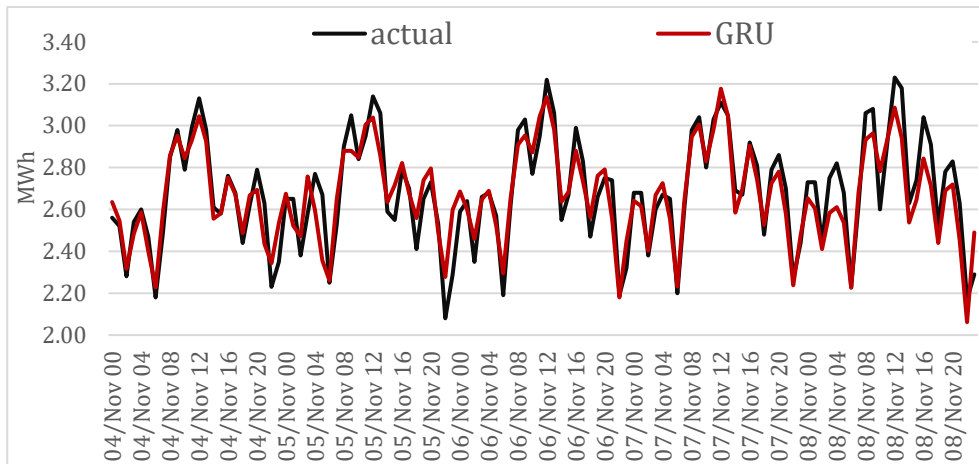


Figura 7.5 Consum real vs prognozat - GRU.

Rețeaua LSTM-GRU îmbunătățește erorile generale, din cauza stratului GRU, dar există un model clar de variații mari pentru valorile de vârf și perioadele de vârf. Autorii din [47] au găsit o diferență clară între LSTM și GRU și sugerează că selecția tipului de unitate recurentă cu încadrare depinde de setul de date și de sarcina corespunzătoare. În cazul studiului de caz prezentat în teza de față, cu prognoza curbelor de sarcină non-casnice, rezultatele sunt clar în favoarea GRU.

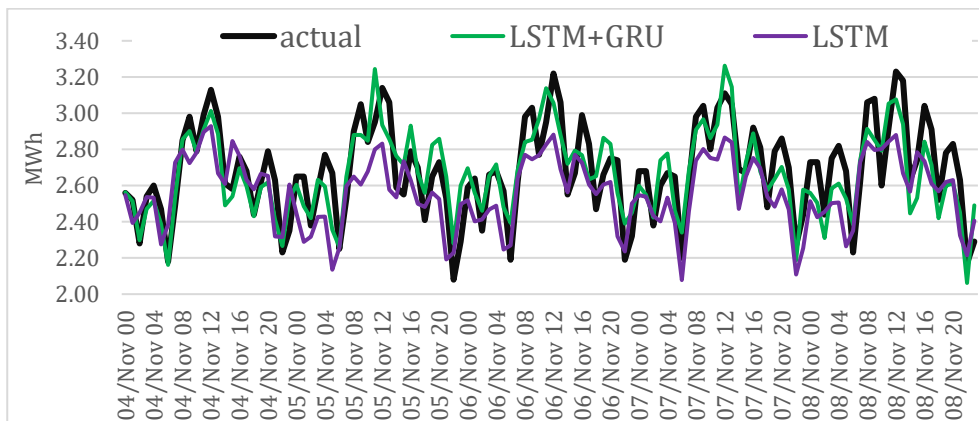


Figura 7.6 Consum real vs prognozat - LSTM-GRU și LSTM.

Pentru o mai bună înțelegere a evoluției rezultatelor GRU și LSTM pentru perioada de testare din Figura 7.7, curba de sarcină reală și cea prognoțată poate fi corelată cu MAPE zilnic.

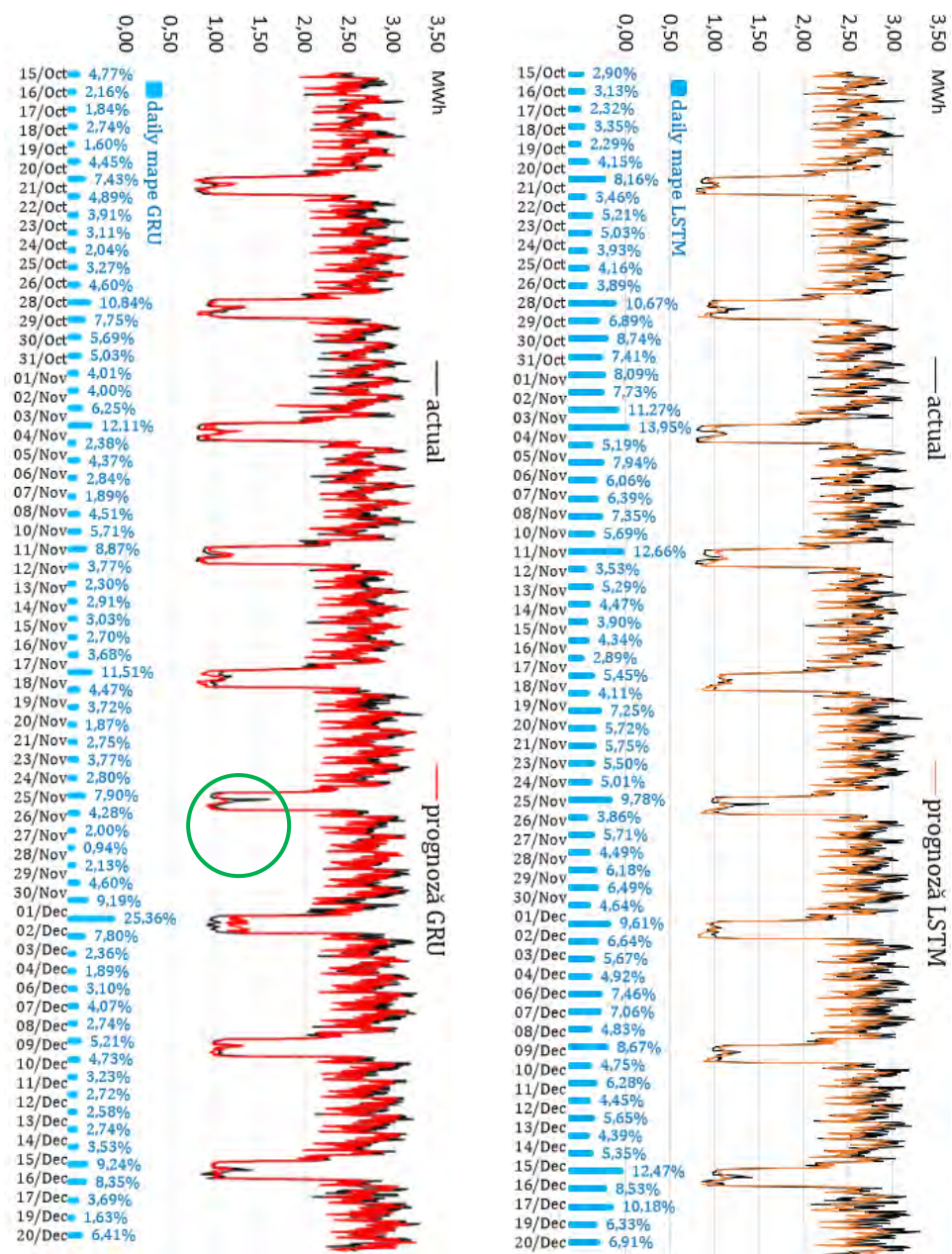


Figura 7.7 MAPE zilnic - GRU și LSTM pentru consumatorul FF

Cel mai bun MAPE zilnic este de 1,58%, iar cea mai mare valoare a erorii este de 25,38%. Eroarea mare a fost cauzată de două perioade de vacanță de la 30 noiembrie până la 1 decembrie 2019. GRU nu reușește să identifice corect evoluția consumului real deoarece în setul de date de

antrenament o perioadă similară nu este „văzută” de rețea, motiv pentru care GRU nu poate interpreta o asemenea situație. Pe deasupra, în weekendul precedent, a avut loc un eveniment neobișnuit cel mai probabil din cauza sistemului de aer comprimat (evidențiat pe figură cu chenar verde). Rezultatele prognozelor aplicate pe curbele de sarcină individuale cu metodele tradiționale prezentate în această teză sunt prezentate în Figura 7.8.

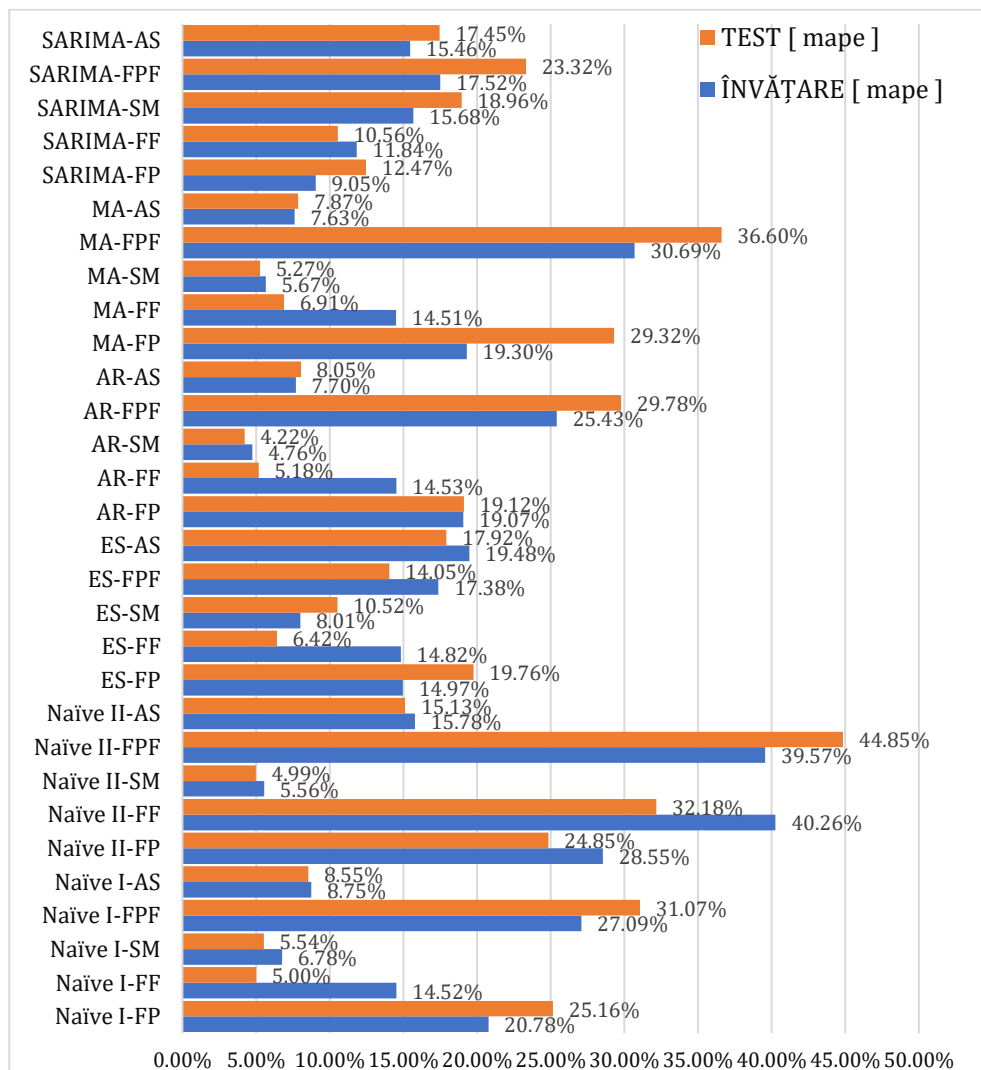


Figura 7.8 Prognoze tradiționale individuale

Curba de sarcină cel mai greu de prognozat aparține FP (fabrica de parchet), aspect datorat volatilității comportamentului de consum.

Curba de sarcină cel mai ușor prognozabilă este cea a supermarket-ului (SM), datorită caracterului repetitiv în timp și influența directă a temperaturii. Spre deosebire de un consumator comercial, care are un program strict de funcționare și un flux tehnologic standardizat, un consumator industrial își schimbă comportamentul de consum în funcție de mulți factori externi și uneori impredictibili (revizii neprogramate, defecțiuni, erori umane etc). Rezultatele prognozelor aplicate pe curbele de sarcină individuale cu metodele ML prezentate în această teză sunt prezentate în Figura 7.9.

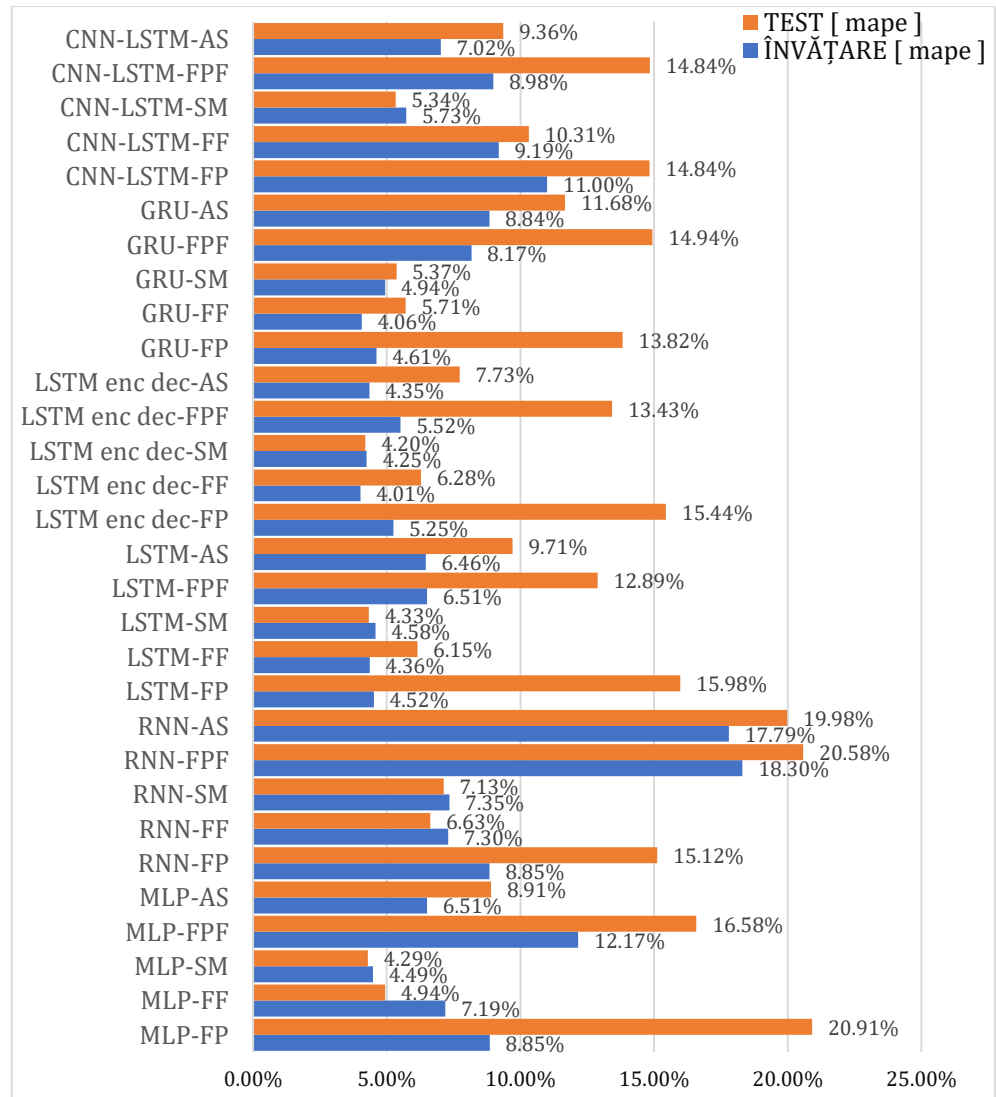


Figura 7.9 Prognoze cu ML individuale

Se pot observa valori mari pentru prognoza consumatorului FP (fabrică de parchet), explicația erorilor de peste 10% MAPE obținută de toți algoritmi este variația imprevizibilă a consumului (Figura 7.10). Fabrica este compusă din trei hale de producție și o clădire de birouri, cu un proces tehnologic complex, iar toată fabrică intră în revizie două sau trei săptămâni pe an. Perioada de revizie tehnică influențează foarte mult algoritmi cu învățare automată, deoarece sunt configurați pentru a modela această perioadă care este foarte volatilă. Având în vedere că perioada de antrenare este din ianuarie până în octombrie, prognozele pentru perioada de testare vor fi puternic influențate de ultimele variații în consum.

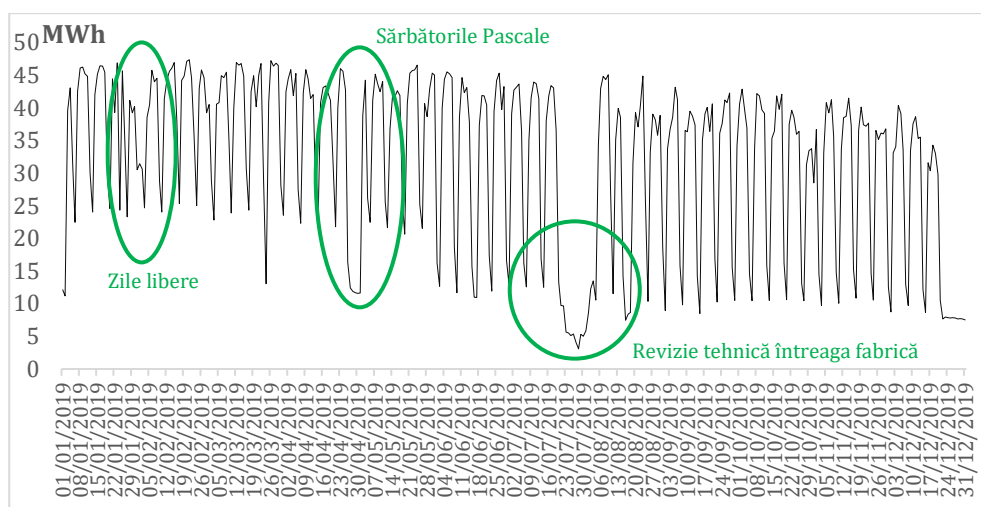


Figura 7.10 Consum de electricitate fabrică producție parchet (PF)

O componentă importantă a tezei constă în compararea prognozelor realizate individual cu prognoza agregată. În Figura 7.11 sunt prezentate erorile MAPE ale celor două abordări și se poate observa îmbunătățirea erorilor în cazul aplicării metodelor de prognoză pe curba de sarcină agregată. Prognoza efectuată individual și ulterior însumată este notată cu $\sum Indiv$ și este descrisă de relația (7.1):

$$\sum Indiv = \hat{Y}_{FP} + \hat{Y}_{FF} + \hat{Y}_{SM} + \hat{Y}_{FPF} + \hat{Y}_{AS} \quad (7.1)$$

S-a considerat această abordare pentru a replica practicile de pe piața de energie electrică pentru furnizorii care obligatoriu sunt parte responsabilă cu echilibrarea (PRE) sau fac parte dintr-un PRE mai mare.

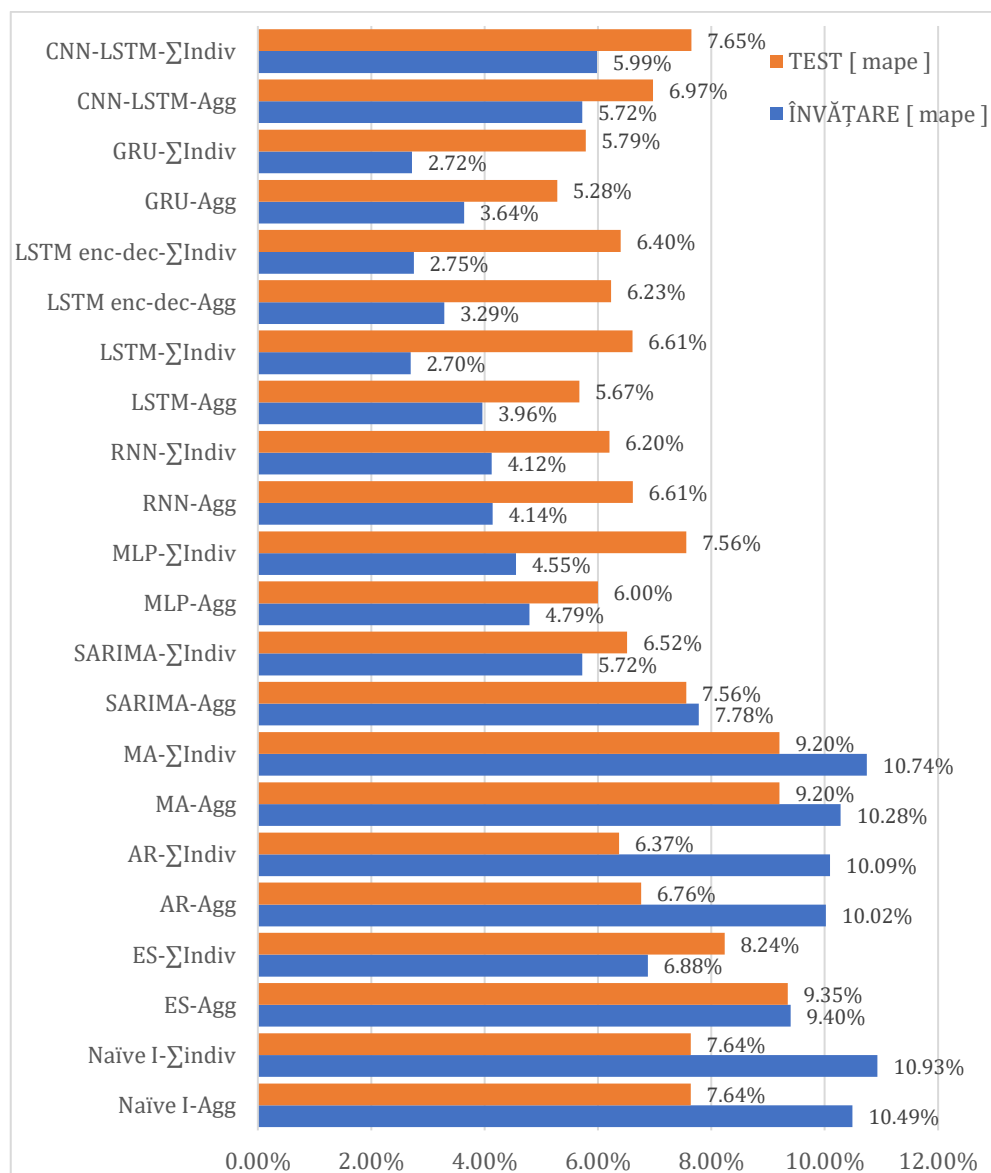


Figura 7.11 Rezultate comparative pentru diferitele metode aplicate

8. Evaluare prognoză pe piața de energie electrică

8.1. Introducere

Evaluarea riscului pe piața de energie este o activitate complexă ce presupune cunoștințe în domeniul energetic și comercial. În această secțiune se simulează cel mai defavorabil scenariu în care furnizorul realizează tranzacții pe piața spot (PZU) și trebuie să gestioneze riscul produs de volatilitatea ridicată a evoluției prețurilor pe această piață [184]. Pentru stabilizarea procesului de tranzacționare este necesar un instrument de prognoză robust.

Rezultatele evaluării implică un grad mare de responsabilitate influențând decizii care pot să determine succesul strategiei aplicate atât la nivelul consumatorului cât și a furnizorului. Pentru ambele părți implicate este necesară o strategie managerială eficientă care să cuprindă atât programe cât și proceduri de gestionare a dezechilibrelor care vizează minimizarea probabilității producerii acestor dezechilibre și al expunerii potențiale [185]. Scopul principal al metodei prezentată în acest capitol este de a oferi un indicator al erorilor prognozei care sintetizează pe date reale, atât ale consumatorilor cât și prețuri din piață. Astfel se urmărește minimizarea pierderilor sau cheltuielilor suplimentare suportate de companii. Spre exemplu, dacă un consumator industrial își oprește producția pentru o zi și nu notifică furnizorul, acesta va achiziționa energie pe baza tiparelor de consum anterioare (prognoze naive) și astfel rezultă o cantitate de energie în exces care trebuie vândută pe piața de echilibrare (PE) la prețul de excedent.

Determinarea riscului potențial pentru un furnizor activ pe piața de energie electrică reprezintă o activitate importantă. Această perspectivă poate fi împărțită în două tipuri de riscuri [186]: riscul comercial datorat nerealizării prognozei de consum a energiei electrice și riscul financiar datorat amplificării costurilor prin tranzacționarea unor mari cantități de energie pe PZU și PE.

Abordarea tezei corelează valorile erorilor calculate (MAPE) pentru fiecare metodă cu eficiența utilizării prognozelor de către un furnizor în contextul pieței de energie electrică, ținând cont de cadrul real în care acesta poate să achiziționeze sau să vândă energie electrică. Se realizează mai multe simulări care modelează expunerea pe PZU și PE pentru perioada în care s-au testat algoritmi de prognoză.

Scenariul pe baza căruia se construiește modul de evaluare este achiziția de pe PZU a energiei electrice pe baza prognozelor realizate, iar diferența care poate să fie surplus sau deficit, este echilibrată pe PE. Participarea pe PE este obligatorie, orice PRE este echilibrat pe această piață, singura excepție este atunci când nu există deloc diferență între prognoză și consum real. Echilibrarea se referă la balanța între cantitatea de energie achiziționată și cea vândută. Piața de echilibrare are două prețuri diferite în funcție de poziția (deficit sau excedent) în care se află participantul la piața. De aceea este important să se evalueze întreg procesul în ansamblu pentru a evidenția implementarea practică.

Acest scenariu este real pentru un furnizor de energie din România, dar un portofoliu care este format dintr-un mix de energie este mult mai complex și diversificat. Diferența dintre valorile prognozate și cele reale, $d_h = \hat{y}_h - y_h$ va determina dezechilibrul orar al cantităților care vor fi achiziționate sau vândute pe PE. Dacă $d_h > 0$, ceea ce înseamnă că prognoza este mai mare decât valorile reale, rezultă un surplus de energie care trebuie vândut pe PE. Dacă $d_h < 0$, ceea ce înseamnă că prognoza este mai mică decât valorile reale, rezultă un deficit care trebuie achiziționat de la PE. De obicei, energia cumpărată de pe piața de echilibrare (PE) este foarte scumpă, iar vânzarea se face la prețuri foarte mici sau negative. Prețurile luate în considerare pentru evaluarea prognozelor sunt prețurile stabilite de către operatorul pieței de energie electrică (OPCOM) în perioada de testare (15 octombrie 2019 - 20 decembrie 2019).

8.2. Ipoteza de lucru

Perioada de testare a algoritmilor cu învățare automată este identică cu perioada în care se implementează evaluarea prognozelor pe piețele PZU și PE. Erorile obținute în perioada de testare sunt corelate cu valori financiare pentru fiecare algoritm utilizat, inclusiv pentru metodele tradiționale. Deoarece datele de consum pentru întreg grupul de colaboratori au fost obținute pentru anul 2019, evaluările efectuate sunt aplicate pe prețurile din perioada menționată pentru testare. Între timp, începând cu 1 februarie 2021 baza de decontare în România a fost modificată la 15 minute și aplicată pentru PE prin Ordinul 213/25.11.2020¹⁴. Având în vedere prevederile Articolului 52 din Regulamentul (UE) 2017/2195 al Comisiei din 23 noiembrie 2017 de

¹⁴ <https://portal.anre.ro/PublicLists/Ordin>

stabilire a unei linii directoare privind echilibrarea sistemului de energie electrică ce a intrat în vigoare la data de 18 decembrie 2017, în cadrul ședinței Comitetului de reglementare al ANRE din data de 25.11.2020 a fost aprobat Regulamentul de calcul și de decontare a dezechilibrelor părților responsabile cu echilibrarea – preț unic de dezechilibru¹⁵. De asemenea ANRE instituie reguli de tranzacționare pe piețele centralizate la termen de energie electrică aprobate prin ordine, în care referirea la durata de o oră să fie înlocuită cu referirea la durata intervalului de decontare, iar această durată a intervalului de decontare să fie de o oră până la data de 1 iulie 2021, respectiv de 15 minute, începând cu data de 1 iulie 2021.

Această modificare a fost necesară ca urmare a aplicării începând cu data de 1 februarie 2021 a intervalului de decontare a pieței de echilibrare și a dezechilibrelor părților responsabile cu echilibrarea (PRE) de 15 minute, pentru a permite participanților la piață să realizeze o poziție cât mai avantajoasă în piața de echilibrare, având la dispoziție intervale de tranzacționare pe piețele la termen egale ca durată cu intervalul de decontare a dezechilibrelor PRE.

Având în vedere faptul ca legislația a fost aprobată abia în februarie 2021 după mai multe amânări și datele de consum obținute de la colaboratori sunt pe anul 2019 s-a ținut cont în lucrare de structura pieței PE și prețurile de la nivelul anului 2019. Metodologia aplicată în lucrare se aplică și pe noua legislație.

Diferențele de energie între cantitățile reale (curbe de sarcină măsurate) și cele contractante sunt de două tipuri:

- a. deficit - cantitatea reală consumată depășește cantitatea contractată;
- b. surplus - cantitatea reală consumată depășește cantitatea contractată.

Chiar dacă cele două situații sunt diferite, efectele financiare produse sunt la fel de puternice, în ambele cazuri putând aduce furnizorului pierderi însemnate, deoarece cantitățile rezultate ca excedent sau deficit trebuie tranzacționate pe piețe extrem de volatile (Figura 8.1), cu fluctuații mari ale prețurilor și puțin controlabile de către furnizor (PZU și PE).

¹⁵<https://www.anre.ro/ro/presa/comunicate/comunicat-privind-aprobarea-regulamentului-de-calcul-si-de-decontare-a-dezechilibrelor-partilor-responsabile-cu-echilibrarea-pret-unic-de-dezechilibru>

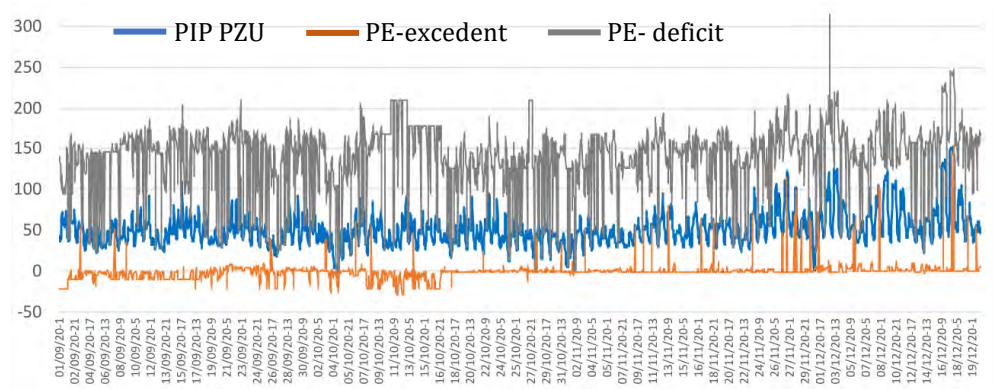


Figura 8.1 Prețuri orare euro/MWh pentru PZU și PE¹⁶

Proгноza realizată cu diversele metode propuse devine astfel cantitate achiziționată de pe platforma PZU. Se calculează dezechilibrul d_t și se determină surplusul sau deficitul de energie electrică (orar).

$$Q_{PZU} = \hat{y}_t \quad (8.1)$$

$$d_t = Q_{PZU} - Y_{actual} = \begin{cases} Q_{PEs}, & \text{if } d_t > 0, & \text{excedent} \\ Q_{PEd}, & \text{if } d_t < 0, & \text{deficit} \\ 0, & \text{if } d_t = 0, & \end{cases} \quad (8.2)$$

unde, Q_{PZU} este cantitatea de energie oferită pe PZU, Q_{PEs} este cantitatea de energie care rămâne în excedent, Q_{PEd} este cantitatea de energie care rămâne în deficit. Pe un interval orar dezechilibrul reprezintă deficit sau excedent, în funcție de diferența dintre prognoza și consum real. Pentru a calcula costul total de tranzacționare pentru un furnizor în scenariul în care utilizează doar PZU și PE ne folosim de ecuația (8.3) care se calculează la nivel de interval de decontare id :

$$C_{id} = Q_{PZU,id} P_{PZU,id} - Q_{PEs,id} P_{PEs,id} + Q_{PEd,id} P_{PEd,id} \quad (8.3)$$

unde $P_{PZU,id}$, $P_{PEs,id}$, $P_{PEd,id}$ reprezintă prețurile orare pentru PZU, preț de excedent și deficit aferente pieței de echilibrare. Pe un interval id

¹⁶ Prețuri publicate de operatorul pieței de energie din România. cursul de schimb pentru conversia în euro în T2 2019 este de 4,7665 lei / euro. * <https://www.bnro.ro/Raport-statistic-606.aspx>.

cantitatea de energie poate să fie deficit sau excedent, în cazul în care $Q_{PEs,id} > 0$ înseamnă că $Q_{PEd,id} = 0$ și invers.

Participarea furnizorilor la PZU și PE are importante consecințe financiare asupra prețului orar de achiziție al energiei electrice, având în vedere că rezultatele înregistrate pe piața PZU și valorile energiei de dezechilibru (excedent sau deficit) variază de la oră la oră în limite foarte largi. În condițiile liberalizării complete a pieței de energie electrică, toți consumatorii au posibilitatea de a-și alege furnizorul. Din punct de vedere al furnizorilor, acest fapt conduce la necesitatea agregării unui număr din ce în ce mai mare și mai diversificat de structuri orare de consum. Însă unul din rolurile unui furnizor este acela de a valorifica avantajele agregării consumului, furnizorii îndeplinind, pentru clienții săi, rolul de agregatori de sarcină (agregatori ai prognozelor elaborate de aceștia).

8.3. Rezultate

Teza își propune să demonstreze caracterul practic al învățării automate pentru a obține previziuni pe termen scurt pentru consumatorii industriali și comerciali agregați. Prognozele sunt validate prin utilizarea indicatorilor clasici pentru erori și prin calcularea impactului pieței pentru ziua următoare și a pieței de echilibrare. Abordarea nouă constă în determinarea impactului economic al previziunilor. Costurile prezentate sunt calculate pe oră și se adună pentru întreaga perioadă de testare (15 octombrie 2019 - 20 decembrie 2019). Diferența financiară între metoda de prognoză cu cele mai bune rezultate față de metoda cu cele mai slabe rezultate este de ~40.044,0 euro pentru întreg cluster-ul de consumatori prognozați în perioada de testare. Această concluzie evidențiază o îmbunătățire a costului cu 8,51% în scenariul prezentat. Comparând cele mai bune două metode, diferența este de 5.326 euro, ceea ce înseamnă că folosind GRU reducem costurile cu 1,24% pentru perioada de testare față de LSTM. Timp de un an, aceste costuri pot fi de patru ori mai mari în funcție de evoluția prețurilor pieței energiei electrice.

În Figura 8.2 se sintetizează rezultatele obținute de fiecare metodă de prognoză aplicată pentru următoarele 24 de ore. Impactul financiar al dezechilibrelor generate de prognoze este prezentat în comparație cu indicatorul MAPE. Se prezintă atât costul total (expunerea pe PZU și PE) cât și costul de echilibrare. Participarea pe PE este obligatorie pentru toți furnizorii, iar costul de echilibrare este generat de

achiziția și vânzarea pe PE, care este determinată de diferența dintre cantitățile de energie achiziționate de pe PZU (prognoze) și consumul real. Indicatorul MAPE nu oferă o delimitare clară între metode, dar în momentul în care se analizează impactul financiar se observă că rezultă o mai bună apreciere a performanțelor metodelor.

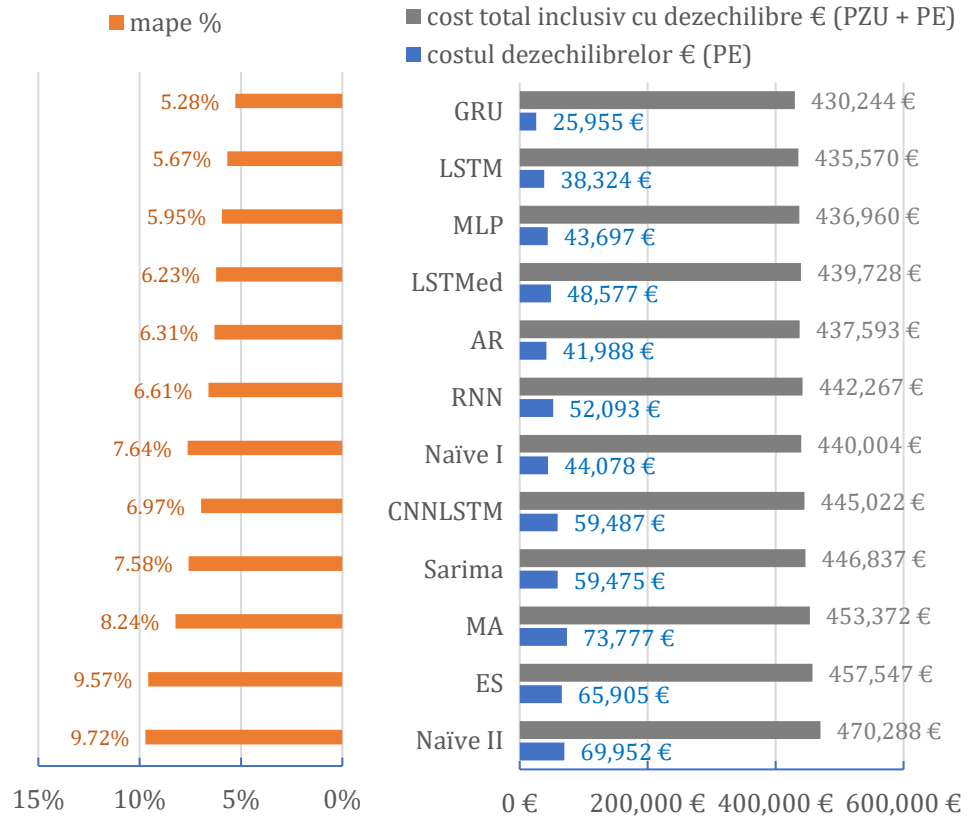


Figura 8.2 Comparația MAPE cu costul erorilor pentru prognoza pe 24 ore.

În Tabel 8.1 se prezintă numeric valorile dezechilibrelor și procentul acestora în costul total de achiziție pentru clusterul de consumatori analizați pentru perioada de testare (15 octombrie– 20 decembrie 2019). Se observă o relație direct proporțională între MAPE și procentul dezechilibrelor în costul de achiziție.

Tabel 8.1 Evaluarea prognozelor cu impactul financiar pe piața de energie – 24 ore

metodă prognoză	costul generat de dezechilibru €	mape %	costul total €	% dezechilibre in costul total de achiziție
Naive II	69.952,2	9.82%	470.288,1	15.70%
ES	65.905,1	9.57%	457.547	14.40%
MA	73.776,58	8.24%	453.371,5	16.27%
SARIMA	59.475,03	7.58%	446.837	13.31%
CNNLSTM	59.487,18	6.97%	445.021,8	13.37%
Naive I	44.078,43	6.80%	440.003,6	10.02%
RNN	52.092,9	6.61%	442.267,3	11.78%
AR	41.988,43	6.31%	437.593,2	9.60%
LSTMed	48.576,9	6.23%	439.727,7	11.05%
MLP	43.697,32	5.95%	436.959,7	10.00%
LSTM	38.324,26	5.67%	435.569,7	8.80%
GRU	25.955,13	5.28%	430.243,5	6.03%

Ținând cont de același raționament aplicat pentru orizontul de 24 de ore, s-au implementat metodele prezentate în Figura 8.3 pentru orizontul de 48 de ore. S-au utilizat metodele care au oferit cele mai bune rezultate pentru 24 ore și s-a implementat o rețea neuronală compusă dintre un strat GRU și unul LSTM, care depășește performanța obținută cu GRU.

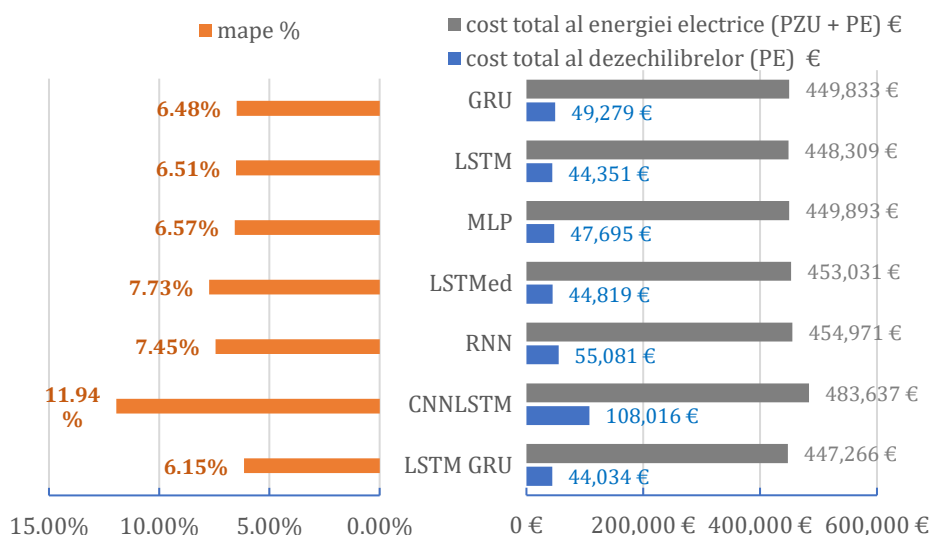


Figura 8.3 Comparația MAPE cu costul erorilor pentru prognoza pe 48 ore.

În Tabel 8.2 se prezintă numeric valorile dezechilibrelor și procentul în costul total de achiziție pentru clusterul de consumatori analizați pentru perioada de testare (15 octombrie– 20 decembrie 2019). Având în vedere orizontul mai mare prognozat, erorile cresc. În consecință crește costul cu dezechilibrele și costul total de achiziție.

Tabel 8.2 Evaluarea prognozelor cu impactul financiar pe piața de energie – 48 ore

	LSTMGRU	CNNLSTM	RNN	LSTMed	MLP	LSTM	GRU
cost total al dezechilibrelor €	44.033,9	108.016,2	55.080,6	44.819,01	47.694,56	44.350,6	49.279,36
MAPE %	6,15%	11,94%	7,45%	7,73%	6,57%	6,51%	6,48%
cost total al energiei electrice (DAM + BM) €	447.266,2	483.636,8	454.971	453.030,8	449.893,3	448.309	449.833
% dezechilibre în costul total de achiziție	9,85%	22,33%	12,11%	9,89%	10,60%	9,89%	10,96%

Pentru evaluarea prognozelor pe orizontul de 168 ore în Figura 8.4 și Tabel 8.3 se prezintă rezultatele financiare. Metoda cu cel mai bun rezultat este combinația între GRU și LSTM, obținând indicatorul MAPE cu 0,53% mai mic decât prin metoda GRU. Orizontul de 168 de ore reprezintă o prognoză realizată orar pentru o săptămână, ceea ce este o informație foarte valoroasă pentru un furnizor de energie, deoarece se pot accesa și alte produse ale pieței de energie pentru minimizarea costurilor și optimizarea achiziției sau vânzării de energie electrică. În contextul decontării la 15 minute, orizontul de 168 de pași devine o prognoză pe două zile. Având în vedere creșterea numărului de pași ai prognozei se observă că a crescut și eroarea, dar rămâne în continuare sub 6%.

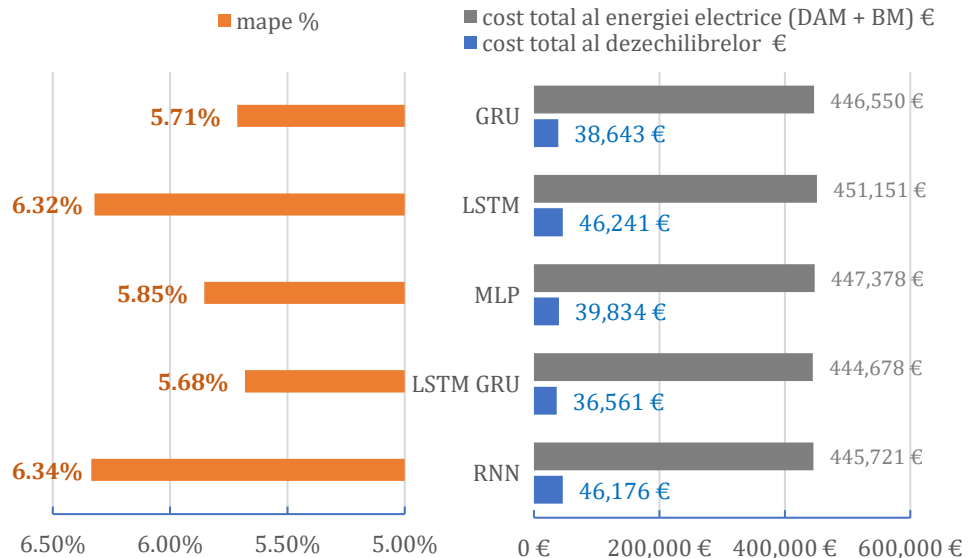


Figura 8.4 Comparația MAPE cu costul erorilor pentru prognoza pe 168 ore.

Tabel 8.3 Evaluarea prognozelor cu impactul financiar pe piața de energie – 168 ore

	RNN	LSTM GRU	MLP	LSTM	GRU
cost total al dezechilibrelor €	31.176,13	33.560,87	39.833,65	46.240,66	38.642,89
mape %	6,34%	5,68%	5,85%	6,32%	5,71%
cost total al energiei electrice (DAM + BM) €	445.720,8	444.677,6	447.378,1	451.150,6	446.550,4
% dezechilibre în costul total de achiziție	6,99%	7,55%	8,90%	10,25%	8,65%

Pentru evidențierea performanțelor metodelor cu cele mai bune rezultate se prezintă în Figura 8.5 o analiză la nivel zilnic a indicatorului MAPE și a costurilor cu dezechilibru pentru prognoza pe 24 de ore a curbei de sarcină agregată. Variația zilnică a costurilor depinde în primul rând de prețurile de pe PZU și PE. În data de 03 Decembrie se poate observa cel mai mare cost cu dezechilibru chiar dacă valoarea MAPE este de 8%, iar prin comparație în data de 30 Noiembrie MAPE este de 16% și practic se obține profit pe PE. Explicația pentru această situație este prețul de excedent care variază și poate să se apropie de valoarea energiei de pe PZU (Figura 8.1). În acest caz furnizorul cumpără mai multă energie de pe PZU (datorită prognozelor) și vinde excedentul la prețul de pe PE. Din punct de vedere al costurilor de echilibrare se obține profit, dar în mixul de achiziție al energiei va rezulta o pierdere deoarece se cumpără la preț PZU și se vinde la un preț de excedent inferior prețului PZU.

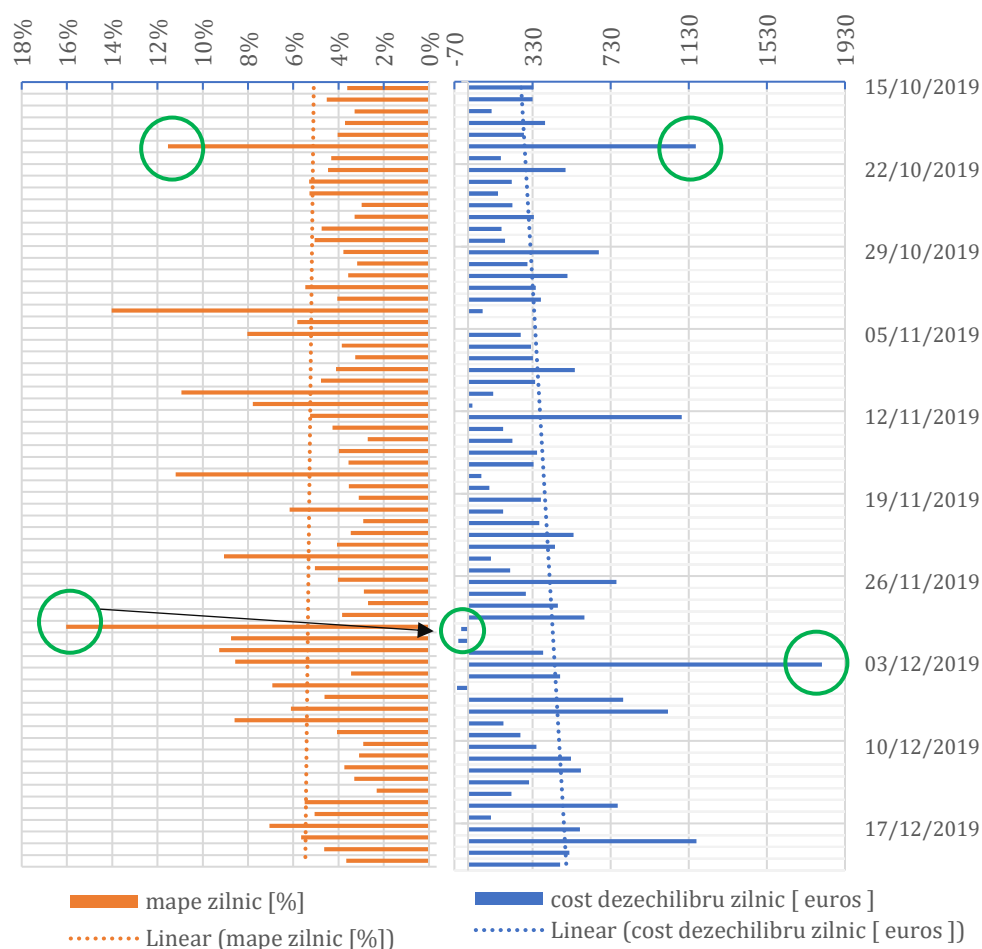


Figura 8.5 Evaluarea zilnică a erorilor MAPE obținute cu GRU (24 ore)

Prețul de excedent poate să fie și negativ, ceea ce înseamnă că furnizorul plătește operatorul de sistem să preia excedentul de energie. Astfel de situații sunt greu de anticipat, de aceea în teză s-au folosit ultimele trei luni din 2019 pentru testare cu prețurile reale din piața de energie.

În Figura 8.6 se prezintă rezultatele zilnice pentru prognoza realizată cu metoda LSTM aplicată pe curba de sarcină agregată pentru 24 de ore și se poate observa o variație diferită a costurilor față de metoda GRU, diferență datorată erorilor generate de prognoză. În aceste demersuri primul pas este determinarea diferenței între prognoză și consum real, din care rezultă poziția de surplus sau excedent al furnizorului. Pe interval de decontare furnizorul poate să fie în excedent

sau deficit, dar în decursul unei zile dezechilibrul se compune atât din cantități de deficit cât și cantități de excedent în funcție de prognoză. În data de 30 Noiembrie consumul real orar a fost mai mic decât prognoza, ceea ce înseamnă un surplus de energie (mai multă energie a fost cumpărată de pe PZU) care se vinde pe PE. Acesta este motivul pentru care în ciuda erorii foarte mari rezultă un mic profit pe PE, dar analizând tot procesul de achiziție rezultă o pierdere, după cum a fost discutat la metoda GRU.

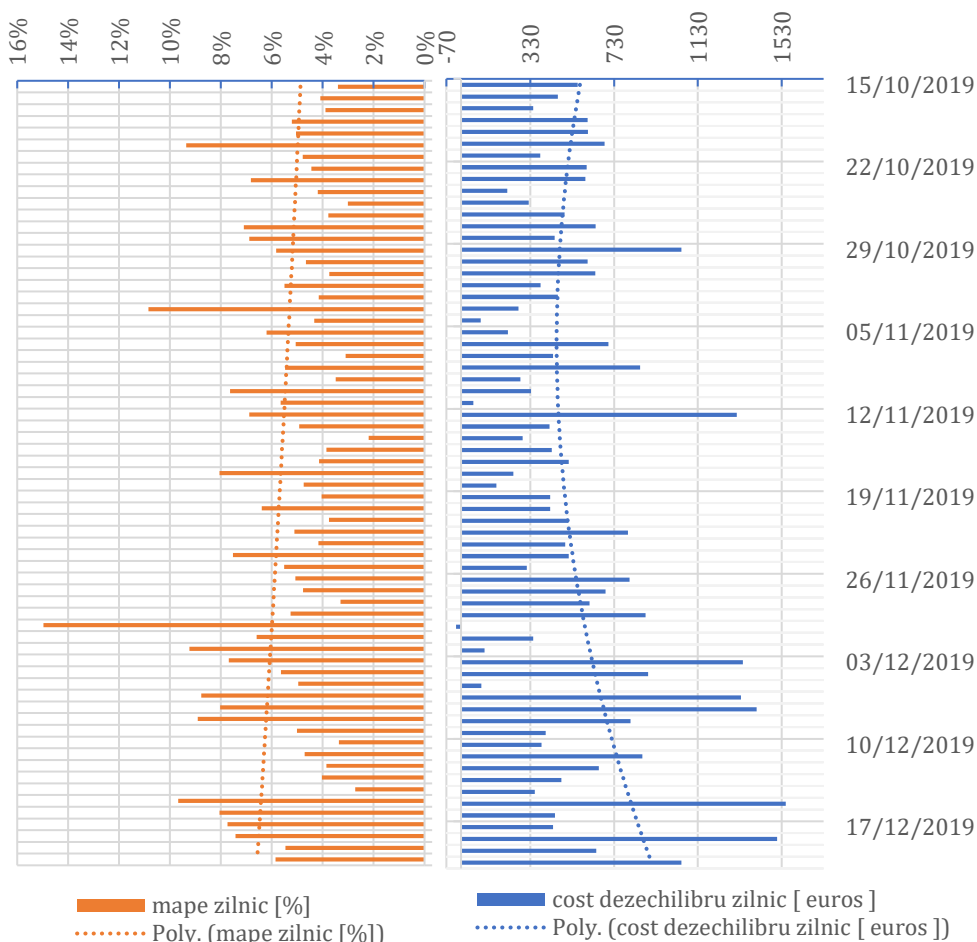


Figura 8.6 Evaluarea zilnică a erorilor MAPE obținute cu LSTM (24 ore)

9. Discuții generale

Majoritatea studiilor din literatura de specialitate cu privire la prognozarea curbelor de sarcină pe termen scurt se adresează consumului rezidențial sau agregat (la nivelul stațiilor electrice, regional sau național). În teza de față se aplică și se compară metode tradiționale cu metode bazate pe învățare automată (ML) pentru prognoza unui grup de 5 consumatori industriali și comerciali care însumează la nivel anual un consum de 37,92 GWh. Din cercetarea literaturii, nu s-au identificat studii care să cuantifice impactul prognozei pe piața energiei electrice și să coreleze erorile prognozei în termeni financiari. Teza de doctorat elaborată oferă o analiză sintetică a literaturii relevante, implementează și compară diverse metode pentru prognozarea curbelor de sarcină nerezidențiale din punct de vedere practic. Metodele de prognoză sunt implementate și comparate pentru curbele de sarcină ale grupului țintă atât individual și agregat pentru a reproduce activitatea unui furnizor de energie pe piața energiei electrice. Această abordare poate ajuta furnizorii de energie să-și optimizeze portofoliul, iar PRE-urile (parțile responsabile cu echilibrarea sau agregatori) pot implementa strategii bazate pe prognoza consumului pentru echilibrarea portofoliului de clienți [187]. Se propune o abordare nouă pentru evaluarea erorilor de prognoză pe baza costurilor generate pe piețele de energie.

Istoricul cu curbele de sarcină este împărțit în două seturi de date (80% antrenare și 20% testare) pentru antrenarea și validarea corectă a algoritmilor cu învățare automată. Datele de testare nu au fost folosite în antrenarea algoritmilor și ajută la identificarea performanței algoritmilor în situații reale de prognoză. Se compară performanțele prognozelor între douăsprezece metode și s-au testat multiple combinații pentru variabilele independente utilizate în prognoză. Rezultatele obținute pe eșantionul de testare pentru 1608 de valori orare (15 octombrie - 20 decembrie 2019) indică în mod constant că: (1) rețelele neuronale recurente sunt potrivite pentru consumul de sarcină nerezidențială; (2) este utilă corelarea erorilor statistice ale prognozei cu costul generat în cazul aplicării prognozei pe instrumentele de tranzacționare ale pieței de energie electrică; (3) prognozele realizate pentru o curbă de sarcină agregată obțin rezultate mai bune decât prognozele individuale realizate pentru fiecare consumator; (4) metodele cu învățare automată obțin rezultate îmbunătățite față de metodele tradiționale; (5) cel mai bun model implementat pentru

curbele de sarcină agregată este GRU și construirea unui model complex (rețea neuronală mare) nu garantează performanța algoritmului.

În partea de evaluare a rezultatelor studiului, este propusă o nouă abordare de evaluare pentru valorile prognozate pentru diferite orizonturi de timp (24, 48, 168 ore) pe istoricul de test. Impactul financiar al erorilor este calculat orar pe baza prețurilor reale din perioada de test aferente pieței pentru ziua următoare (PZU) și a pieței de echilibrare (PE). Valorile prognozate reprezintă notificări de ofertă orare pentru PZU. Diferența dintre valorile reale și cele prognozate sunt decontate obligatoriu la prețurile de deficit și excedent, evidențiind impactul financiar orar. Acest cost orar este comparat cu indicatori statistici pentru erori (MAPE, MAE, RMSE), astfel se poate cuantifica impactul erorilor și se evidențiază importanța prognozei. Costurile prezentate sunt calculate la nivel orar pentru perioada de testare.

Pentru prognoza pe 24 de ore a curbei agregate aferente grupului de consumatori analizați, diferența între valoarea totală a costului (pe perioada de testare) dintre algoritmul cu cea mai mică eroare de prognoză (GRU – MAPE de 5,28%) și metoda cu cea mai mare eroare (Naivă II – MAPE de 9,28%) este de 40.044,0 euro. Rezultă o îmbunătățire cu 8,51% a costurilor totale de achiziție pentru întreg grupul de consumatori pentru cele 3 luni de testare. Comparând cele mai bune două metode, diferența este de 5.326 euro, ceea ce înseamnă că folosind GRU în detrimentul LSTM, reducem costurile cu 1,24% pentru perioada de testare. Timp de un an, aceste costuri pot fi de patru ori mai mari în funcție de evoluția prețurilor pe piața de energie electrică.

Comparând cea mai bună metodă de prognoză (GRU– MAPE de 5,28%) cu cea mai bună abordare naivă (Naivă II – MAPE 6,8%), se obține o diferență de cost de 9.760,0 euro pentru cele trei luni în care s-au testat modelele. Metoda Naivă II reprezintă o prognoză empirică în care ziua precedentă devine prognoza pentru ziua următoare, cu mențiunea că se ține cont de zilele de weekend și sărbători. Cea mai bună metodă de prognoză tradițională (AR– MAPE de 6,31%) obține un cost mai ridicat decât GRU cu 7.349,0 euro. Diferența dintre GRU și a doua cea mai bună metodă (LSTM – MAPE de 5,6%) este de 5326,17 euro. Noutatea adusă în cadrul tezei este aplicarea și evaluarea algoritmilor de învățare automată pentru prognoza de consum a unui grup agregat de consumatori industriali și comerciali, ținând cont de activitatea unui furnizor de energie în contextul actual al pieței. Utilizarea algoritmilor cu învățare automată și instrumente de programare care permit prelucrarea datelor în timp real este un prim pas spre implementarea

conceptului de rețele electrice inteligente (smart grid) la consumatori. Motivul pentru utilizarea metodelor cu învățare automată este complexitatea și volumul uriaș de date care necesită gestionare automată pentru a lua decizii rapide și corecte.

Trebuie menționat că aceste costuri sunt calculate pentru întreg grupul de consumatori și în prezent sunt suportate de furnizor și consumator, care prin prognoze eficiente pot să minimizeze acest cost. Furnizorul nu poate cumpăra energie de pe piață individual pentru fiecare consumator în parte, el are la dispoziție două metode din care rezultă cantitățile orare pe baza cărora se cumpără energie: i) primește sau realizează prognoze individuale pentru fiecare client care sunt însumate; ii) se însumează datele acualizate de consum de la clienți și se realizează prognoza curbei de sarcină agregată. În această teză se implementează și se compară ambele cazuri.

Istoricul utilizat în această teză este întreg anul 2019, deoarece datele obținute de la toți consumatorii analizați se suprapun în această perioadă. Între timp, de la jumătatea anului 2021, Autoritatea Națională de Reglementare în domeniul Energiei (ANRE) a aprobat modificarea intervalului de decontare la 15 min pentru piața de echilibrare, iar prețurile la energie spre finalul lui 2021 au crescut de aproape 4 ori față de media din 2019. Această creștere nu poate fi explicată de trecerea la decontarea la 15 minute, deoarece contextul pieței de energie este mult mai complex, dar cu siguranță cresc costurile cu dezechilibre suportate de furnizori. Pentru a veni în întâmpinarea acestei situații, în teză s-a implementat prognoza pe 168 de pași, ceea ce reprezintă o prognoză pe 42 de ore efectuată la fiecare 15 minute. În acest mod furnizorul poate folosi prognoza pentru achiziția de energie pe PZU și notificările realizate la 15 minute pentru PE.

Dezvoltarea viitoare a acestei lucrări va extinde orizontul de prognoză la mai mulți pași pentru a permite integrarea pretului de pe piața intra-zilnică, decontarea în 15 minute pe piața de echilibrare și evaluarea prognozelor ținând cont și de alte instrumente financiare disponibile pe piața de energie. Principala limitare în prognoza curbelor de sarcină este accesul la datele de consum actualizate. Toate datele utilizate în această analiză sunt colectate de la contoarele operatorilor de distribuție și obținute de la furnizori. Acești consumatori nu au sisteme de monitorizare a energiei, ceea ce ar ajuta foarte mult la prognoză. O altă direcție de cercetare este utilizarea contoarelor "inteligente" la consumatorii analizați în această lucrare, comunicarea în timp real a datelor de consum și realizarea prognozei în timp real.

Din punct de vedere al pieței, consumatorii de energie electrică nerezidențială din România nu sunt conștienți de impactul previziunilor sau al modelelor de consum asupra facturilor lor de energie electrică. Există și cazuri în care furnizorii penalizează consumatorii care depășesc o marjă de variație a consumului față de prognoză, dar majoritatea consumatorilor de energie electrică din România au un preț unic pe MWh. Consecința este că industria nu programează cu atenție producția și nu comunică activ cu furnizorul. O posibilă reglementare utilă ar fi promovarea tarifelor diferențiate orar, stimularea consumatorului să urmeze un model de consum și comunicarea activă cu furnizorul în cazul schimbărilor majore în producție. Creșterea programelor de eficiență energetică și promovarea sistemelor de monitorizare inteligente cu comunicații în timp real au potențialul să sprijine integrarea consumatorilor în rețele electrice inteligente.

10. Concluzii finale

10.1. Concluzii generale

Prognoza consumului de energie electrică are un impact deosebit asupra operării și funcționării sistemului energetic național, atât tehnic cât și financiar. Precizia prognozei de consum determină funcționarea tehnică a SEN care determină costul suportat de participanții la piața de energie. Punctul de plecare este prognoza de consum, pe baza căreia se realizează planificarea resurselor. Implicarea activă a consumatorilor, alături de furnizori pentru realizarea prognozelor de consum este necesară pentru minimizarea dezechilibrelor. Inacuratețea prognozei de consum determina o serie de costuri suplimentare care afectează în mod direct activitatea comercială a furnizorului și consumatorului, iar indirect a întregului SEN.

Pentru evidențierea importanței prognozei de consum, în această teză se reproduce activitatea unui furnizor de energie care achiziționează energie pentru un portofoliu de consumatori industriali și comerciali. Achiziția se realizează la prețurile de pe piața spot (PZU) și piața de echilibrare (PE). Diferitele metode de prognoză implementate sunt comparate și evaluate în funcție de costurile generate de erorile obținute. Având în vedere faptul că s-au realizat prognoze pe diferite orizonturi de timp, se menționează că aceste metode pot fi folosite și pentru diferitele posibilități de achiziție a energiei de pe piețele existente, luând în calcul scenarii care să cuprindă atât contracte bilaterale cu achiziție în bandă, vârf și gol a energiei, cât și tranzacționarea energiei pe PZU, respectiv PE.

În paragrafele următoare se sintetizează rezultatele obținute în această lucrare:

- s-au analizat și implementat diverși algoritmi cu învățare automată (MLP, RF, RNN, LSTM, GRU, LSTM “encoder-decoder”, LSTM-CNN și LSTM-GRU) în comparație cu metode tradiționale (RL, RML, AR, MA, SARIMA, NE și două metode naive).
- setul de date (istoricul) este împărțit în 80% din date pentru antrenare și 20% din date pentru testarea algoritmilor. În perioada de antrenare se determină parametrii (ponderi și erori aleatorii în rețelele neuronale) algoritmilor cu învățare automată și coeficienții metodelor statistice. Perioada de testare este folosită doar pentru evaluarea metodelor, nu se actualizează parametrii algoritmilor.

- algoritmi cu învățare automată și SARIMA sunt implementați în Python utilizând editorul Jupyter, platforma Tensorflow și librăriile Keras. Sunt utilizate și alte instrumente pentru prelucrare și vizualizare date menționate în capitolele anterioare. Metodele tradiționale implementate RL, RML, AR, MA, NE și cele două metode naive sunt implementate în Excel folosind instrumentele de analiză statistică disponibile (Data Analysis și Solver).
- motivele pentru alegerea acestor instrumente de programare sunt: i) multitudinea de opțiuni pentru încărcarea, prelucrarea și vizualizarea datelor (fișiere .csv); (ii) facilitează implementarea algoritmilor cu învățare automată; și (iii) perspectivele pentru dezvoltarea și implementarea metodelor în timp real în aplicații cloud pentru grupuri de consumatori.
- algoritmi cu învățare automată sunt antrenați în mod similar utilizând aceleași variabile independente, aceeași funcție cost (MSE), aceeași rată de învățare și algoritmul gradient descendent cu optimizare ADAM, toate fiind configurabile prin intermediul librăriilor Keras. Numărul de straturi ascunse, numărul neuroni în fiecare strat și numărul de epoci variază deoarece s-a urmărit obținerea celor mai bune rezultate. O rețea neuronală recurentă necesită mai multe epoci pentru antrenare decât o rețea perceptron multistrat, datorită numărului mai mare de parametri (ponderi).
- rezultatele cele mai bune (erori minime) s-au obținut utilizând algoritmi cu învățare automată bazați pe rețele neuronale recurente - respectiv GRU și LSTM. Eroarea cea mai mică fiind obținută de GRU cu MAPE de 5,28% pentru curbe agregate și MAPE de 4,20% pentru curbe de sarcină individuale (consumatorul supermarket - SM). S-au obținut rezultate diferite în funcție de curbele de sarcină prognozate. Curba de sarcină a unui supermarket are o variație zilnică repetitivă, datorită fluxului tehnologic bine structurat, motiv pentru care toate metodele de prognoză au obținut erori mici. Eroarea obținută cu metoda autoregresivă (AR) este MAPE de 4,22% și cu GRU se obține MAPE de 4,37%.
- în contrast cu curbele de sarcină ale supermarket-ului, fabrica de parchet are un comportament de consum mult mai variabil de la o zi la alta și prin consecință cea mai mică eroare de prognoză s-a obținut cu GRU având un MAPE de 13,82%. O explicație pentru aceste rezultate este perioada de revizie de patru săptămâni din iulie în care consumul se reduce la 25% din medie și este foarte variabil (utilajele

principale sunt recondiționate și testate în sarcină, iar instalația de aer comprimat funcționează într-un regim intermitent).

- algoritmi ML au un grad ridicat de stocasticitate (ponderile și bias-ul sunt inițiate aleator) ceea ce generează rezultate diferite după rulări succesive – în această lucrare se prezintă cele mai bune rezultate pentru toate metodele, o valoare de referință este GRU cu MAPE de 5,28% pentru curbele de sarcină agregate. Algoritmi ML obțin erori (în medie) cu 2% MAPE mai mici decât algoritmi tradiționali. Pentru prognoza pe 24 ore a curbei agregate, eroarea obținută cu metoda GRU este cu 1,03% MAPE mai mică decât eroarea obținută de cea mai bună metodă tradițională (AR – MAPE de 6,31%), ceea ce înseamnă o îmbunătățire procentuală a erorii de 16,3%.
- pentru identificarea celor mai bune arhitecturi pentru rețelele neuronale s-au implementat o gamă largă de configurații și simulări, cele mai reprezentative sunt prezentate pentru algoritmul GRU și se pot studia în Anexe.
- cu cât rețelele neuronale sunt mai complexe (adânci) cu un număr mare de straturi și neuroni, eroarea MAPE pentru prognoză crește atât pentru antrenare cât și pentru testare. Acest aspect este evidențiat printr-un coeficient de complexitate DL_{index} , care cuantifică dimensiunea rețelei și numărul de epoci (numărul de ponderi este determinat de numărul de straturi ascunse și numărul de neuroni din fiecare strat). Dimensiunea rețelei se obține prin apelarea unei funcții aferente instrumentului Tensorflow.
- trebuie menționat că aceste rezultate sunt limitate de resursele hardware folosite: PC Intel(R) Core(TM) i5-4690K CPU@3.5GHz, RAM 16GB, 64-bit sistem de operare, x64-procesor. Prin folosirea unui CPU sau GPU mai puternic este posibil să se obțină rezultate mai bune cu rețele mai complexe. Referitor la resursele hardware și timpul de învățare, în teză se pune accentul pe componenta practică a prognozei ceea ce înseamnă limitarea timpului de antrenare. Prognozele trebuie realizate rapid pentru a putea fi încărcate în timp util pe platformele de tranzacționare a energiei electrice. Este necesar un timp de maxim 30 de minute pentru antrenare și prognozare pentru a putea încărca ofertele pe PZU și să se considere ultimele date de consum.
- s-a antrenat o rețea GRU (24|4|744|744|744|48|24) cu ~5 milioane de parametri pentru 10, 30, 50, 100, 150, 200 epoci, și s-a obținut cea mai bună eroare MAPE (la antrenarea de 50 epoci) pentru perioada de testare de 5,43%. Pe lângă faptul că se obține o eroare mai mare

decât GRU (24|3|100|100|48|24) timpul de antrenare ajunge până la 26 de ore.

- rezultatele sunt cuantificate în termeni financiari, practic s-a simulat situația unui furnizor care cumpăra sau vinde energie pe baza prognozelor pe piața de energie. Se transformă erorile obținute în costuri dacă s-ar utiliza prognozele pentru achiziția de energie. Metoda GRU este mai bună decât LSTM cu 5.326,0 euro pe perioada de testare din 15 octombrie până în 20 decembrie 2019 (folosind prețuri și condiții de piață reale).
- comparând cea mai bună metodă de prognoză cu învățare automată (GRU- MAPE de 5,28%) cu cea mai bună de prognoză tradițională (AR- MAPE de 6,31%) se obține o reducere a costurilor prin utilizarea GRU cu 7.349,0 euro.
- pe măsură ce orizontul de prognoză crește, întâlnim erori mai mari pentru prognoză și în consecință costuri ridicate.
- toate calculele pentru erori și costuri de dezechilibru sunt efectuate orar pentru întreaga perioadă de testare, dar s-au evidențiat acești indicatori și la nivel zilnic.
- aceste analize pot să reprezinte argumente solide în încercarea de a explica necesitatea prognozelor reprezentanților companiilor private. Astfel se utilizează un limbaj comun și direct pentru a înțelege efectele unui comportament volatic energetic, spre deosebire de comunicarea unei erori de 5,28% MAPE. Astfel putem să corelăm MAPE cu riscul, iar în cazul prognozei pe 24 de ore a grupului de consumatori prezentat în această teză pe perioada de testare rezultă că 0,5% MAPE \approx 5.000,0 euro. Abordarea agregată reduce costurile cu 5.000,0 euro în comparație cu abordarea individuală.
- analizând rezultatele se stabilește GRU ca fiind cea mai bună metodă atât pentru prognoza individuală pe 24 de ore (MAPE de 5,79%) cât și cea agregată (MAPE de 5,28%). Prognozarea agregată obține un rezultat cu un MAPE de 0,51% mai bun decât abordarea individuală, ceea ce înseamnă o îmbunătățire procentuală cu 8,81% și un cost redus cu \sim 5.000,0 euro.
- abordarea agregată reduce costurile furnizorului și ale consumatorului, este evident că aceasta nu ajută la operarea rețelei electrice din punct de vedere tehnic. Abordarea individuală are nevoie de monitorizare inteligentă pentru a putea fi utilizată cu succes pentru toate entitățile implicate. Accesul la date în timp real,

prelucrarea și aplicarea algoritmilor de prognoză automată necesită infrastructură care momentan nu există. De aceea, în această teză se prezintă situația pieței de energie electrică actuală. Se poate menționa că abordarea individuală are inițial nevoie de rețele electrice inteligente, pentru ca apoi să poată optimiza funcționarea rețelei electrice.

Furnizorii sunt interesați ca clienții lor să folosească tot mai multă energie, dar prețurile mari la energie, dezvoltarea sistemelor inteligente de monitorizare, implementarea programelor de control a curbelor de sarcină (demand response) și a rețelei inteligente [119] [120] vor schimba status quo-ul definit de un comportament iresponsabil către eficientizarea activă a consumului de energie. Realizarea prognozei consumului de energie este facilitată de aceste tehnologii și la rândul ei poate ajuta la ridicarea gradului de automatizare și optimizare a funcționării rețelei electrice. Teza se adresează consumului de electricitate din sectorului industrial și comercial, care conform autorilor din [121], reprezintă 63,46% din consumul total de energie electrică din lume. Provocarea este de a anticipa comportamentul consumatorilor mari pentru a putea gestiona resursele energetice eficient. Această teză demonstrează că prognozarea curbelor de sarcină agregate oferă erori și costuri mai mici pe piața energiei electrice, dar abordarea responsabilă ar fi prognozarea fiecărui consumator pe baza proceselor tehnologice specifice monitorizate adecvat. Limitarea prognozării individuale este lipsa contorizării inteligente și a senzorilor. Pe baza audit-ului energetic realizat la fabrica de parchet, rezultă un contur energetic care cuprinde uscătoarele de lemn tip depozit. Consumul uscătoarelor de energie este influențat direct de temperatura și umiditatea exterioară, dar datorită faptului că datele de consum obținute sunt doar de pe conturul total atunci nu s-a putut identifica o corelație puternică pentru prognoză. Fără date relevante, modelele de prognoză eficiente și replicabile nu reprezintă o investiție fiabilă pentru sectorul privat. Chiar dacă din punct de vedere financiar este logic să se prevadă curbe de sarcină agregate, din perspectiva tehnică a rețelelor electrice, este de preferat să se prognozeze consumatorii individuali pentru a îmbunătăți și funcționarea rețelei electrice.

Apropierea dintre consumator și rețele electrice inteligente înseamnă gestionarea proceselor industriale în mod optim pentru a elimina excesul de consum, gestionarea unităților proprii de producție a energiei sau participarea în programe de reducere a consumului pentru a reduce vârful de sarcină sau de a vinde excesul de energie. Aceste

operațiuni se doresc a fi realizate automat în timp real și prin comunicații în două sensuri între operatorul de sistem și consumator pentru a opera fluxul de energie cât mai eficient posibil și cu cât mai puține pierderi. Aplicațiile industriale au cerințe diferite față de consumatorii rezidențiali, energia fiind consumată de o gamă largă de dispozitive și în cantități mult mai mari. Procesele tehnologice deja existente în exploatare necesită funcționarea și coordonarea cu sisteme centralizate de control printr-o gamă largă de standarde / protocoale cu un nivel ridicat de interacțiune.

Consumatorii industriali variază de la rafinăriile de petrol și fabrici chimice până la fabrici cu linii de asamblare pentru automobile sau utilități cu instalații de prelucrare a produselor alimentare. În afară de a fi mari consumatori de energie electrică, mulți dintre acești consumatori au capacități de generare proprii și pot fi producători de energie electrică. Conceptul de integrare a consumatorilor industriali în rețele inteligente se bazează pe interoperabilitate și interacțiunea dintre rețeaua electrică și instalații industriale, inclusiv producerea de energie electrică la nivelul consumatorului. Transferul de energie are loc între instalațiile industriale și rețeaua electrică, prin intermediul unei interfețe programate corespunzător, cu scopul de a satisface cererea fluctuantă de energie la o calitate înaltă și preț previzibil.

10.2. Originalitatea și contribuțiile inovative ale tezei

10.2.1. Distincția față de literatura anterioară

Pornind de la stagiul actual de dezvoltare și studierea tendințelor în industria emergentă a rețelelor electrice inteligente, cercetarea s-a concentrat pe idei noi și pe analize practice realizate în colaborare cu mai mulți consumatori industriali din sectorul industrial și comercial. Provoacă această teză este de a dezvolta algoritmi capabili să “învețe” tipare de consum și să prognozeze cu erori minime diferite orizonturi de timp (1, 24, 48, 168 ore). Noutatea tezei este aplicarea algoritmilor cu învățare automată pentru prognoza curbelor de sarcină aferente unui grup de consumatori industriali și comerciali. Se propune o abordare nouă pentru evaluarea erorilor de prognoză pe baza costurilor generate de erorile de prognoză prin reproducerea activității unui furnizor de energie pe piața de energie electrică. Evaluarea dezechilibrului dintre valorile prognozate și reale se face cu indicatori statistici și calculul impactului financiar al erorilor de prognoză în mediul pieței de energie. Pentru perioada de testare modelul simulează achiziția de energie pe piața pentru ziua următoare (PZU) și expunerea generată de erorile de prognoză în funcție de prețurile de deficit și surplus aferente pieței de echilibrare (PE).

Motivul pentru care se utilizează algoritmi cu învățare automată pentru prognoză este performanța ridicată și faptul că facilitează aplicabilitatea în mediul privat. Algoritmii sunt implementați în medii de programare open source și au capacitatea să automatizeze procesul de prognoză. Având în vedere volumul mare de date și necesitatea gestionării acestora în timp real, algoritmi trebuie să se ajusteze parametrii necesari învățării conform noilor date de consum cu minimă intervenție umană. S-au implementat în total douăsprezece metode de prognoză pentru compararea metodelor tradiționale cu metodele bazate pe învățare automată.

Pentru implementarea algoritmilor cu învățare automată s-au realizat o serie de simulări pentru identificarea celor mai bune arhitecturi și configurări. Se evidențiază problema supra-adaptării și sub-adaptării rețelelor neuronale prin definirea unui index de complexitate al rețelelor neuronale și compararea acestuia cu evoluția erorii MAPE în funcție de numărul de epoci și timpul de antrenare.

Acest studiu oferă o analiză detaliată a literaturii de specialitate, implementează și compară diverse metode ML pentru prognozarea curbelor de sarcină industriale și comerciale din punct de vedere practic. Metodele de prognoză sunt implementate și comparate pentru curbele de sarcină individuale și agregate pentru a sublinia impactul financiar asupra pieței energiei electrice. Această abordare poate ajuta furnizorii de energie să-și optimizeze portofoliul, iar strategii DR (demand response) pot fi implementate pe baza prognozelor de consum.

Lucrarea se concentrează mai mult pe rezultate și încearcă să găsească cea mai bună soluție la problemele de prognoză în ceea ce privește viteza, acuratețea și simplitatea.

10.2.2. Limitări și studii viitoare

Istoricul de consum este disponibil pentru un an întreg (2019) pentru toți consumatorii. Seturile de date mai mari nu sunt accesibile momentan pentru fiecare consumator. Datele de pe piața energiei electrice utilizate în etapa de evaluare sunt disponibile online pe site-ul operatorului pieței energiei electrice [69]. Prețurile energiei electrice sunt extrem de volatile și au un impact puternic asupra portofoliilor de echilibrare a furnizorilor de energie. Prognozarea acestor prețuri este greu de realizat și ar introduce un grad în plus de incertitudine, motiv pentru care în teză se utilizează istoricul pentru evidențierea unor situații reale întâlnite în practică. Extinderea studiului ar trebui să cuprindă perioade mai lungi cu date istorice. Trebuie menționat faptul că nu s-au putut obține date de la consumatori în 2020 datorită pandemiei care a determinat oprirea completă a fabricii de parchet pentru trei luni, unitatea de procesare a cărnii s-a oprit complet din activitate, iar restul consumatorilor au funcționat intermitent. În aceste condiții nu s-a putut asigura continuitatea datelor pentru întreg grupul țintă și s-au luat în calcul datele complete din anul 2019.

Implementarea propusă utilizează prețurile orare de pe piața de energie pentru trei luni pentru a valida caracterul practic al prognozelor. Perioadă care se consideră suficientă, dat fiind faptul că se poate lucra cu mai mulți consumatori. În 2019 decontarea energiei electrice se realiza la interval orar, dar din mijlocul anului 2021, autoritatea în domeniul energiei din România (ANRE) a aprobat modificarea intervalului de decontare la 15 minute, ceea ce crește costul dezechilibrului. Această

modificare impune ca notificările efectuate pe piața de echilibrare să se bazeze pe prognoze efectuate la 15 minute. Chiar dacă nu a fost abordată direct această modificare în teză, s-a efectuat prognoza pe un orizont de 168 de pași care acoperă un orizont de 42 de ore prognozate la fiecare 15 minute.

Limitarea principală în prognozarea consumului este accesul la date. Toate datele utilizate în această analiză sunt colectate de la contoare de distribuție și obținute de la furnizori. Acești consumatori nu au sisteme de monitorizare a energiei care ar ajuta foarte mult la prognoză. De exemplu, procesarea alimentelor, depozitarea cărnii sunt procese tehnologice influențate de temperatură, în timp ce producția de carne depinde de productivitate, iar analiza separată ar putea crește precizia de prognozare.

Parametrii rețelelor neuronale sunt configurați manual în etapa de antrenament pentru a obține cele mai bune rezultate pe setul de testare, sarcină care necesită mult timp. Deoarece se poate obține o precizie ridicată asupra setului de antrenament, obiectivul nostru este să dezvoltăm modele care să reducă la minimum erorile pentru datele de testare, luând în considerare utilitatea practică a prognozelor. Metodele ML utilizate sunt antrenate pe date până la sfârșitul lunii septembrie și prognozează restul anului folosind același model antrenat, fără a adapta parametrii rețelelor cu datele viitoare. Pentru studii ulterioare se urmărește configurarea automată a parametrilor algoritmilor cu învățare automată printr-un proces de antrenare mai complex și adaptarea actualizată a parametrilor.

Se urmărește dezvoltarea de produse online pentru prognoză adaptivă pentru consumul energiei electrice, definite de cerințele de piață și focusate pe direcțiile de implementare a rețelelor electrice inteligente.

REFERINTE

- [1] E. commission, A european Green Deal [Online], (n.d.). https://ec.europa.eu/info/strategy/priorities-2019-2024/european-green-deal_en.
- [2] N. Iqtiyaniillham, M. Hasanuzzaman, M. Hosenuzzaman, European smart grid prospects, policies, and challenges, *Renew. Sustain. Energy Rev.* 67 (2017) 776–790. <https://doi.org/https://doi.org/10.1016/j.rser.2016.09.014>.
- [3] D. Fan, Y. Ren, Q. Feng, Y. Liu, Z. Wang, J. Lin, Restoration of smart grids: Current status, challenges, and opportunities, *Renew. Sustain. Energy Rev.* 143 (2021) 110909. <https://doi.org/https://doi.org/10.1016/j.rser.2021.110909>.
- [4] J.A.P. Lopes, A.G. Madureira, M. Matos, R.J. Bessa, V. Monteiro, J.L. Afonso, S.F. Santos, J.P.S. Catalão, C.H. Antunes, P. Magalhães, The future of power systems: Challenges, trends, and upcoming paradigms, *WIREs Energy Environ.* 9 (2020) e368. <https://doi.org/https://doi.org/10.1002/wene.368>.
- [5] G. F., V. J., C. F., M. A., F. G., Smart grid projects outlook 2017: facts, figures and trends in Europe, (2017). <https://doi.org/https://doi:10.2760/15583>.
- [6] R. Zhang, Y. Du, L. Yuhong, New challenges to power system planning and operation of smart grid development in China, in: 2010 Int. Conf. Power Syst. Technol., 2010: pp. 1–8. <https://doi.org/10.1109/POWERCON.2010.5666114>.
- [7] I. Alotaibi, M.A. Abido, M. Khalid, A. V Savkin, A Comprehensive Review of Recent Advances in Smart Grids: A Sustainable Future with Renewable Energy Resources, *Energies.* 13 (2020). <https://doi.org/10.3390/en13236269>.
- [8] C. Eid, P. Codani, Y. Perez, J. Reneses, R. Hakvoort, Managing electric flexibility from Distributed Energy Resources: A review of incentives for market design, *Renew. Sustain. Energy Rev.* 64 (2016) 237–247. <https://doi.org/10.1016/j.rser.2016.06.008>.
- [9] A.F. Meyabadi, M.H. Deihimi, A review of demand-side management: Reconsidering theoretical framework, *Renew. Sustain. Energy Rev.* 80 (2017) 367–379. <https://doi.org/https://doi.org/10.1016/j.rser.2017.05.207>.
- [10] S. Woltmann, A. Cordes, M. Stomberg, J. Kittel, Using Multi-Agent Systems for Demand Response Aggregators: A Technical Implementation, in: 2020 25th IEEE Int. Conf. Emerg. Technol. Fact. Autom., 2020: pp. 911–918. <https://doi.org/10.1109/ETFA46521.2020.9212168>.
- [11] Y.M. Ding, S.H. Hong, X.H. Li, A demand response energy management scheme for industrial facilities in smart grid, *IEEE Trans. Ind. Informatics.* 10 (2014) 2257–2269. <https://doi.org/10.1109/TII.2014.2330995>.
- [12] R. Lu, R. Bai, Y. Huang, Y. Li, J. Jiang, Y. Ding, Data-driven real-time price-based demand response for industrial facilities energy management, *Appl. Energy.* 283 (2021) 116291. <https://doi.org/https://doi.org/10.1016/j.apenergy.2020.116291>.
- [13] L. Timma, R. Skudritis, D. Blumberga, Benchmarking Analysis of Energy Consumption in Supermarkets, *Energy Procedia.* 95 (2016) 435–438. <https://doi.org/https://doi.org/10.1016/j.egypro.2016.09.056>.
- [14] G. Pellegrini-Masini, A. Pirni, S. Maran, C.A. Klöckner, Delivering a timely and Just Energy Transition: Which policy research priorities?, *Environ. Policy Gov.* 30 (2020) 293–305. <https://doi.org/10.1002/eet.1892>.
- [15] N.E. Koltsaklis, A.S. Dagoumas, G. Seritan, R. Porumb, Energy transition in the

- South East Europe: The case of the Romanian power system, *Energy Reports*. 6 (2020) 2376–2393. <https://doi.org/10.1016/j.egy.2020.07.032>.
- [16] L. Hernandez, C. Baladron, J.M. Aguiar, B. Carro, A.J. Sanchez-Esguevillas, J. Lloret, J. Massana, A Survey on Electric Power Demand Forecasting: Future Trends in Smart Grids, Microgrids and Smart Buildings, *IEEE Commun. Surv. Tutorials*. 16 (2014) 1460–1495. <https://doi.org/10.1109/SURV.2014.032014.00094>.
- [17] F.M. Andersen, H. V Larsen, T.K. Boomsma, Long-term forecasting of hourly electricity load: Identification of consumption profiles and segmentation of customers, *Energy Convers. Manag.* 68 (2013) 244–252. <https://doi.org/https://doi.org/10.1016/j.enconman.2013.01.018>.
- [18] G. Chicco, Overview and performance assessment of the clustering methods for electrical load pattern grouping, *Energy*. 42 (2012) 68–80. <https://doi.org/https://doi.org/10.1016/j.energy.2011.12.031>.
- [19] G. George-Ufot, Y. Qu, I.J. Orji, Sustainable lifestyle factors influencing industries' electric consumption patterns using Fuzzy logic and DEMATEL: The Nigerian perspective, *J. Clean. Prod.* 162 (2017) 624–634. <https://doi.org/10.1016/j.jclepro.2017.05.188>.
- [20] R. Christen., L. Mazzola., A. Denzler., E. Portmann., Exogenous Data for Load Forecasting: A Review, in: *Proc. 12th Int. Jt. Conf. Comput. Intell. - Vol. 1 CI4EMS*, SciTePress, 2020: pp. 489–500. <https://doi.org/10.5220/0010213204890500>.
- [21] J. Sowinski, The Impact of the Selection of Exogenous Variables in the ANFIS Model on the Results of the Daily Load Forecast in the Power Company, *Energies*. 14 (2021). <https://doi.org/10.3390/en14020345>.
- [22] B.Y.G. et al, Machine learning algorithms performed no better than regression models for prognostication in traumatic brain injury, *J. Clin. Epidemiol.* 122 (2020) 95–107. <https://doi.org/https://doi.org/10.1016/j.jclinepi.2020.03.005>.
- [23] C. Kuster, Y. Rezgui, M. Mourshed, Electrical load forecasting models: A critical systematic review, *Sustain. Cities Soc.* 35 (2017) 257–270. <https://doi.org/https://doi.org/10.1016/j.scs.2017.08.009>.
- [24] K.C. Green, J.S. Armstrong, Simple versus complex forecasting: The evidence, *J. Bus. Res.* 68 (2015) 1678–1685. <https://doi.org/https://doi.org/10.1016/j.jbusres.2015.03.026>.
- [25] J.S. Armstrong, K.C. Green, A. Graefe, Golden rule of forecasting: Be conservative, *J. Bus. Res.* 68 (2015) 1717–1731. <https://doi.org/10.1016/j.jbusres.2015.03.031>.
- [26] K. Zor, O. Timur, A. Teke, A state-of-the-art review of artificial intelligence techniques for short-term electric load forecasting, in: *2017 6th Int. Youth Conf. Energy*, 2017: pp. 1–7. <https://doi.org/10.1109/IYCE.2017.8003734>.
- [27] R.J. Hyndman, A brief history of forecasting competitions, *Int. J. Forecast.* 36 (2020) 7–14. <https://doi.org/https://doi.org/10.1016/j.ijforecast.2019.03.015>.
- [28] S. Makridakis, E. Spiliotis, V. Assimakopoulos, The M4 Competition: 100,000 time series and 61 forecasting methods, *Int. J. Forecast.* 36 (2020) 54–74. <https://doi.org/https://doi.org/10.1016/j.ijforecast.2019.04.014>.
- [29] P. Lara-Benítez, M. Carranza-García, J.C. Riquelme, An Experimental Review on Deep Learning Architectures for Time Series Forecasting, *Int. J. Neural Syst.* 31

- (2021) 2130001. <https://doi.org/10.1142/s0129065721300011>.
- [30] J.C. Chambers, S.K. Mullick, and Donald D. Smith, How to Choose the Right Forecasting Technique, *Magazine*. (1971). <https://doi.org/->.
- [31] J.S. Armstrong, Selecting Forecasting Methods, *Princ. Forecast. Int. Ser. Oper. Res. \& Manag. Sci.* 30 (2001) 365–386. https://doi.org/https://doi.org/10.1007/978-0-306-47630-3_16.
- [32] B.H. Archer, Forecasting demand: Quantitative and intuitive techniques, *Int. J. Tour. Manag.* 1 (1980) 5–12. [https://doi.org/https://doi.org/10.1016/0143-2516\(80\)90016-X](https://doi.org/https://doi.org/10.1016/0143-2516(80)90016-X).
- [33] Robert G. BROWN, EXPONENTIAL SMOOTHING FOR PREDICTING DEMAND., Philip Morris Records; Master Settlement Agreement, 1956. <https://www.industrydocuments.ucsf.edu/docs/jzlc0130>.
- [34] C.C. Holt, Forecasting seasonals and trends by exponentially weighted moving averages, *Int. J. Forecast.* 20 (2004) 5–10. <https://doi.org/https://doi.org/10.1016/j.ijforecast.2003.09.015>.
- [35] P.R. Winters, Forecasting Sales by Exponentially Weighted Moving Averages, *Manage. Sci.* 6 (2021) 324–342. <https://doi.org/https://doi.org/10.1287/mnsc.6.3.324>.
- [36] J.W. Taylor, P.E. McSharry, Short-Term Load Forecasting Methods: An Evaluation Based on European Data, *{IEEE} Trans. Power Syst.* 22 (2007) 2213–2219. <https://doi.org/10.1109/tpwrs.2007.907583>.
- [37] R.J. Hyndman, G. Athanasopoulos, *Forecasting: principles and practice*, 2nd edition, (n.d.). OTexts.com/fpp2.
- [38] G.E.P. Box, G.M. Jenkins, *Time Series Analysis: Forecasting and Control*, Holden-Day, 1976.
- [39] L. Suganthi, A.A. Samuel, Energy models for demand forecasting—A review, *Renew. Sustain. Energy Rev.* 16 (2012) 1223–1240. <https://doi.org/https://doi.org/10.1016/j.rser.2011.08.014>.
- [40] T. Fang, R. Lahdelma, Evaluation of a multiple linear regression model and SARIMA model in forecasting heat demand for district heating system, *Appl. Energy.* 179 (2016) 544–552. <https://doi.org/https://doi.org/10.1016/j.apenergy.2016.06.133>.
- [41] S. Ray, A Quick Review of Machine Learning Algorithms, in: *2019 Int. Conf. Mach. Learn. Big Data, Cloud Parallel Comput.*, 2019: pp. 35–39. <https://doi.org/10.1109/COMITCon.2019.8862451>.
- [42] G. Serin, B. Sener, M. Ozbayoglu, H.O. Unver, Review of tool condition monitoring in machining and opportunities for deep learning, *Int. J. Adv. Manuf. Technol.* 109 (2020). <https://doi.org/10.1007/s00170-020-05449-w>.
- [43] A. Aldahiri, B. Alrashed, W. Hussain, Trends in Using IoT with Machine Learning in Health Prediction System, *Forecasting.* 3 (2021) 181–207. <https://doi.org/10.3390/forecast3010012>.
- [44] R. Cioffi, M. Travaglioni, G. Piscitelli, A. Petrillo, F. De Felice, Artificial Intelligence and Machine Learning Applications in Smart Production: Progress, Trends, and Directions, *Sustainability.* 12 (2020). <https://doi.org/10.3390/su12020492>.
- [45] A. Almalaq, G. Edwards, A Review of Deep Learning Methods Applied on Load Forecasting, in: *2017 16th IEEE Int. Conf. Mach. Learn. Appl.*, 2017: pp. 511–516. <https://doi.org/10.1109/ICMLA.2017.0-110>.
- [46] L. Zhang, J. Wen, Y. Li, J. Chen, Y. Ye, Y. Fu, W. Livingood, A review of machine

- learning in building load prediction, *Appl. Energy*. 285 (2021) 116452. <https://doi.org/https://doi.org/10.1016/j.apenergy.2021.116452>.
- [47] A. Mashlakov, T. Kuronen, L. Lensu, A. Kaarna, S. Honkapuro, Assessing the performance of deep learning models for multivariate probabilistic energy forecasting, *Appl. Energy*. 285 (2021). <https://doi.org/10.1016/j.apenergy.2020.116405>.
- [48] G. Gross, F.D. Galiana, Short-term load forecasting, *Proc. IEEE*. 75 (1987) 1558–1573. <https://doi.org/10.1109/PROC.1987.13927>.
- [49] Y. Wang, S. Sun, X. Chen, X. Zeng, Y. Kong, J. Chen, Y. Guo, T. Wang, Short-term load forecasting of industrial customers based on SVM and XGBoost, *Int. J. Electr. Power & Energy Syst.* 129 (2021) 106830. <https://doi.org/https://doi.org/10.1016/j.ijepes.2021.106830>.
- [50] E. Almehaie, H. Soltan, A methodology for Electric Power Load Forecasting, *Alexandria Eng. J.* 50 (2011) 137–144. <https://doi.org/10.1016/j.aej.2011.01.015>.
- [51] Y. Tao, F. Zhao, H. Yuan, C.S. Lai, Z. Xu, W. Ng, R. Li, X. Li, L.L. Lai, Revisit Neural Network based Load Forecasting, in: 2019 20th Int. Conf. Intell. Syst. Appl. to Power Syst. ISAP 2019, 2019. <https://doi.org/10.1109/ISAP48318.2019.9065930>.
- [52] M. DJUKANOVIC, B. BABIC, D.J. SOBAJIC, Y.-H. PAO, Unsupervised/supervised learning concept for 24-hour load forecasting, *IEE Proceedings. Part C. Gener. Transm. Distrib.* (1993).
- [53] D. Upadhaya, R. Thakur, N.K. Singh, A systematic review on the methods of short term load forecasting, in: 2019 2nd Int. Conf. Power Energy Environ. Intell. Control. PEEIC 2019, 2019: pp. 6–11. <https://doi.org/10.1109/PEEIC47157.2019.8976518>.
- [54] M. Cai, M. Pipattanasomporn, S. Rahman, Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques, *Appl. Energy*. 236 (2019) 1078–1088. <https://doi.org/https://doi.org/10.1016/j.apenergy.2018.12.042>.
- [55] J. Moon, J. Park, E. Hwang, S. Jun, Forecasting power consumption for higher educational institutions based on machine learning, *J. Supercomput.* 74 (2018) 3778–3800. <https://doi.org/10.1007/s11227-017-2022-x>.
- [56] W. Guo, L. Che, M. Shahidehpour, X. Wan, Machine-Learning based methods in short-term load forecasting, *Electr. J.* 34 (2021) 106884. <https://doi.org/https://doi.org/10.1016/j.tej.2020.106884>.
- [57] A. Bellahsen, H. Dagdougui, Aggregated short-term load forecasting for heterogeneous buildings using machine learning with peak estimation, *Energy Build.* 237 (2021) 110742. <https://doi.org/https://doi.org/10.1016/j.enbuild.2021.110742>.
- [58] J. Kim, J. Moon, E. Hwang, P. Kang, Recurrent inception convolution neural network for multi short-term load forecasting, *Energy Build.* 194 (2019) 328–341. <https://doi.org/10.1016/j.enbuild.2019.04.034>.
- [59] Y. Hu, J. Li, M. Hong, J. Ren, R. Lin, Y. Liu, M. Liu, Y. Man, Short term electric load forecasting model and its verification for process industrial enterprises based on hybrid GA-PSO-BPNN algorithm—A case study of papermaking process, *Energy*. 170 (2019) 1215–1227. <https://doi.org/https://doi.org/10.1016/j.energy.2018.12.208>.

- [60] H. Eskandari, M. Imani, M.P. Moghaddam, Convolutional and recurrent neural network based model for short-term load forecasting, *Electr. Power Syst. Res.* 195 (2021). <https://doi.org/10.1016/j.epsr.2021.107173>.
- [61] H.-B. Chen, L.-L. Pei, Y.-F. Zhao, Forecasting seasonal variations in electricity consumption and electricity usage efficiency of industrial sectors using a grey modeling approach, *Energy*. 222 (2021) 119952. <https://doi.org/https://doi.org/10.1016/j.energy.2021.119952>.
- [62] E. Yukseltan, A. Yucekaya, A.H. Bilge, Hourly electricity demand forecasting using Fourier analysis with feedback, *Energy Strateg. Rev.* 31 (2020) 100524. <https://doi.org/https://doi.org/10.1016/j.esr.2020.100524>.
- [63] J. Wang, S. Zhu, W. Zhang, H. Lu, Combined modeling for electric load forecasting with adaptive particle swarm optimization, *Energy*. 35 (2010) 1671–1678. <https://doi.org/https://doi.org/10.1016/j.energy.2009.12.015>.
- [64] M. Khashei, F. Chahkoutahi, A comprehensive low-risk and cost parallel hybrid method for electricity load forecasting, *Comput. Ind. Eng.* 155 (2021). <https://doi.org/10.1016/j.cie.2021.107182>.
- [65] J. Zhang, Y.-M. Wei, D. Li, Z. Tan, J. Zhou, Short term electricity load forecasting using a hybrid model, *Energy*. 158 (2018) 774–781. <https://doi.org/https://doi.org/10.1016/j.energy.2018.06.012>.
- [66] P. Singh, P. Dwivedi, A novel hybrid model based on neural network and multi-objective optimization for effective load forecast, *Energy*. 182 (2019) 606–622. <https://doi.org/https://doi.org/10.1016/j.energy.2019.06.075>.
- [67] W. He, Load Forecasting via Deep Neural Networks, *Procedia Comput. Sci.* 122 (2017) 308–314. <https://doi.org/https://doi.org/10.1016/j.procs.2017.11.374>.
- [68] M.N. Fekri, H. Patel, K. Grolinger, V. Sharma, Deep learning for load forecasting with smart meter data: Online Adaptive Recurrent Neural Network, *Appl. Energy*. 282 (2021) 116177. <https://doi.org/https://doi.org/10.1016/j.apenergy.2020.116177>.
- [69] L. Sehovac, K. Grolinger, Deep Learning for Load Forecasting: Sequence to Sequence Recurrent Neural Networks With Attention, *IEEE Access*. 8 (2020) 36411–36426. <https://doi.org/10.1109/ACCESS.2020.2975738>.
- [70] G. Chitalia, M. Pipattanasomporn, V. Garg, S. Rahman, Robust short-term electrical load forecasting framework for commercial buildings using deep recurrent neural networks, *Appl. Energy*. 278 (2020) 115410. <https://doi.org/https://doi.org/10.1016/j.apenergy.2020.115410>.
- [71] H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent Neural Networks for Time Series Forecasting: Current status and future directions, *Int. J. Forecast.* 37 (2021) 388–427. <https://doi.org/https://doi.org/10.1016/j.ijforecast.2020.06.008>.
- [72] A. Marvuglia, A. Messineo, Using Recurrent Artificial Neural Networks to Forecast Household Electricity Consumption, *Energy Procedia*. 14 (2012) 45–55. <https://doi.org/https://doi.org/10.1016/j.egypro.2011.12.895>.
- [73] N. Mughees, S.A. Mohsin, A. Mughees, A. Mughees, Deep sequence to sequence Bi-LSTM neural networks for day-ahead peak load forecasting, *Expert Syst. Appl.* 175 (2021) 114844. <https://doi.org/https://doi.org/10.1016/j.eswa.2021.114844>.
- [74] C.K. Karlo Hainsch Leonard Göke, P.-Y. Oei, C. von Hirschhausen, European

- Green Deal: Using Ambitious Climate Targets and Renewable Energy to Climb out of the Economic Crisis, *DIW Wkly. Rep.* 28/29 / 2020, S. 303-310. (n.d.). https://doi.org/https://doi.org/10.18723/diw_dwr:2020-28-1.
- [75] M. Child, C. Kemfert, D. Bogdanov, C. Breyer, Flexible electricity generation, grid exchange and storage for the transition to a 100% renewable energy system in Europe, *Renew. Energy.* 139 (2019) 80–101. <https://doi.org/10.1016/j.renene.2019.02.077>.
- [76] J.Z. Thellufsen, H. Lund, P. Sorknæs, P.A. Østergaard, M. Chang, D. Drysdale, S. Nielsen, S.R. Djørup, K. Sperling, Smart energy cities in a 100% renewable energy context, *Renew. Sustain. Energy Rev.* 129 (2020). <https://doi.org/10.1016/j.rser.2020.109922>.
- [77] E. O’Shaughnessy, J.R. Cruce, K. Xu, Too much of a good thing? Global trends in the curtailment of solar PV, *Sol. Energy.* 208 (2020) 1068–1077. <https://doi.org/https://doi.org/10.1016/j.solener.2020.08.075>.
- [78] C. Syranidou, J. Linssen, D. Stolten, M. Robinius, On the Curtailments of Variable Renewable Energy Sources in Europe and the Role of Load Shifting, in: *UPEC 2020 - 2020 55th Int. Univ. Power Eng. Conf. Proc.*, 2020. <https://doi.org/10.1109/UPEC49904.2020.9209846>.
- [79] M.C. Bonjean Stanton, S. Dessai, J. Paavola, A systematic review of the impacts of climate variability and change on electricity systems in Europe, *Energy.* 109 (2016) 1148–1159. <https://doi.org/10.1016/j.energy.2016.05.015>.
- [80] M.G. Rasmussen, G.B. Andresen, M. Greiner, Storage and balancing synergies in a fully or highly renewable pan-European power system, *Energy Policy.* 51 (2012) 642–651. <https://doi.org/10.1016/j.enpol.2012.09.009>.
- [81] C. Fant, B. Boehlert, K. Strzepek, P. Larsen, A. White, S. Gulati, Y. Li, J. Martinich, Climate change impacts and costs to U.S. electricity transmission and distribution infrastructure, *Energy.* 195 (2020) 116899. <https://doi.org/https://doi.org/10.1016/j.energy.2020.116899>.
- [82] Piața Angro, (n.d.). <https://www.anre.ro/ro/energie-electrica/legislatie/piata-de-energie-electrica/proceduri-piata-angro> (accessed July 15, 2020).
- [83] I. Report, Continental Europe Synchronous Area Separation on 8 January 2021 About ENTSO-E, 2021. <https://www.entsoe.eu/news/2021/01/15/system-separation-in-the-continental-europe-synchronous-area-on-8-january-2021-update/>.
- [84] T. Ahmad, D. Zhang, C. Huang, H. Zhang, N. Dai, Y. Song, H. Chen, Artificial intelligence in sustainable energy industry: Status Quo, challenges and opportunities, *J. Clean. Prod.* 289 (2021). <https://doi.org/10.1016/j.jclepro.2021.125834>.
- [85] S.-V. Oprea, A. Bâra, B.G. Tudorică, M.I. Călinoiu, M.A. Botezatu, Insights into demand-side management with big data analytics in electricity consumers’ behaviour, *Comput. Electr. Eng.* 89 (2021). <https://doi.org/10.1016/j.compeleceng.2020.106902>.
- [86] S. Wang, X. Sun, J. Geng, Y. Han, C. Zhang, W. Zhang, Application and Analysis of Big Data Technology in Smart Grid, in: *J. Phys. Conf. Ser.*, 2020. <https://doi.org/10.1088/1742-6596/1639/1/012043>.
- [87] Mingming Zhang, K.L. Lo, A comparison of imbalance settlement methods of electricity markets, in: *2009 44th Int. Univ. Power Eng. Conf.*, 2009: pp. 1–5.
- [88] E. (2015) Fraunhofer-ISI, Electricity Costs of Energy Intensive Industries - An

- International Comparison, (n.d.).
<https://www.isi.fraunhofer.de/content/dam/isi/dokumente/ccx/2015/Electricity-Costs-of-Energy-Intensive-Industries.pdf>.
- [89] E.A. Feinberg, D. Genethliou, Load Forecasting, in: J.H. Chow, F.F. Wu, J. Momoh (Eds.), *Appl. Math. Restructured Electr. Power Syst. Optim. Control. Comput. Intell.*, Springer US, Boston, MA, 2005: pp. 269–285. https://doi.org/10.1007/0-387-23471-3_12.
- [90] I.K. Nti, M. Teimeh, O. Nyarko-Boateng, A.F. Adekoya, Electricity load forecasting: a systematic review, *J. Electr. Syst. Inf. Technol.* 7 (2020) 13. <https://doi.org/10.1186/s43067-020-00021-8>.
- [91] K. Tazi, F. Abdi, M.F. Abbou, Demand and Energy Management in Smart Grid: Techniques and Implementation, in: 2017 Int. Renew. Sustain. Energy Conf., 2017: pp. 1–6. <https://doi.org/10.1109/IRSEC.2017.8477305>.
- [92] R. Liu, Y. Liu, Z. Jing, Impact of industrial virtual power plant on renewable energy integration, *Glob. Energy Interconnect.* 3 (2020) 545–552. <https://doi.org/10.1016/j.gloi.2021.01.004>.
- [93] I.E. Agency, World electricity final consumption by sector, 1974-2018, IEA, Paris, (n.d.). <https://www.iea.org/data-and-statistics/charts/world-electricity-final-consumption-by-sector-1974-2018>.
- [94] O. Diaconu, G. Opreescu, R. Pittman, Electricity reform in Romania, *Util. Policy.* 17 (2009) 114–124. <https://doi.org/https://doi.org/10.1016/j.jup.2008.01.010>.
- [95] N.I. of Statistics, National Institute of Statistics, (n.d.). <https://insse.ro/cms/>.
- [96] J.M.K.C.D. et al. (2020), Energy Education - Industrial energy use [Online], (n.d.). https://energyeducation.ca/encyclopedia/Industrial_energy_use#cite_note-OED-1.
- [97] M. Gunsay, C. Bilir, G. Poyrazoglu, Load Profile Segmentation for Electricity Market Settlement, in: 2020 17th Int. Conf. Eur. Energy Mark., 2020: pp. 1–5. <https://doi.org/10.1109/EEM49802.2020.9221889>.
- [98] T.S.H. Moerenhout, S. Sharma, J. Urpelainen, Commercial and industrial consumers' perspectives on electricity pricing reform: Evidence from India, *Energy Policy.* 130 (2019) 162–171. <https://doi.org/https://doi.org/10.1016/j.enpol.2019.03.046>.
- [99] A. Otsuka, An economic analysis of electricity demand: Evidence from Japan, 2017. <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85044517867&partnerID=40&md5=6ea237d8a53fa39a0c34092c5cd65375>.
- [100] F. Pereira, P. Faria, Z. Vale, Definition of distinct consumer modelling approaches for the participation in Demand Response programs considering distributed generation, in: 2015 IEEE PES Innov. Smart Grid Technol. Lat. Am. (ISGT LATAM), 2015: pp. 608–613. <https://doi.org/10.1109/ISGT-LA.2015.7381224>.
- [101] Y. Cheng, Y. Li, Research of Classification of Electricity Consumers Based on Principal Component Analysis, in: 2009 Sixth Int. Conf. Fuzzy Syst. Knowl. Discov., 2009: pp. 201–206. <https://doi.org/10.1109/FSKD.2009.487>.
- [102] A. Gellert, A. Florea, U. Fiore, F. Palmieri, P. Zanetti, A study on forecasting electricity production and consumption in smart cities and factories, *Int. J. Inf. Manage.* 49 (2019) 546–556. <https://doi.org/https://doi.org/10.1016/j.ijinfomgt.2019.01.006>.
- [103] M.R. Braun, H. Altan, S.B.M. Beck, Using regression analysis to predict the future

- energy consumption of a supermarket in the UK, *Appl. Energy*. 130 (2014) 305–313. <https://doi.org/10.1016/j.apenergy.2014.05.062>.
- [104] J.D. Hobby, G.H. Tucci, Analysis of the residential, commercial and industrial electricity consumption, in: 2011 IEEE PES Innov. Smart Grid Technol., 2011: pp. 1–7. <https://doi.org/10.1109/ISGT-Asia.2011.6167087>.
- [105] M. learning libraries, Tensorflow, (n.d.). <https://www.tensorflow.org/>.
- [106] M. learning libraries, Keras, (n.d.). <https://keras.io/>.
- [107] M. learning libraries, scikit-learn, (n.d.). <https://scikit-learn.org/stable>.
- [108] M. learning libraries, Numpy, (n.d.). <https://numpy.org/>.
- [109] M. learning libraries, Matplotlib, (n.d.). <https://matplotlib.org/>.
- [110] M. learning libraries, Seaborn, (n.d.). <https://seaborn.pydata.org/>.
- [111] D.A. Dickey, W.A. Fuller, Distribution of the Estimators for Autoregressive Time Series with a Unit Root, *J. Am. Stat. Assoc.* 74 (1979) 427–431. <https://doi.org/10.1080/01621459.1979.10482531>.
- [112] H. ALEXANDER, I. BARBARA, D. SUSAN, Introductory Business Statistics, OpenStax, n.d. <https://openstax.org/details/books/introductory-business-statistics>.
- [113] Python, Statsmodels, (n.d.). <https://www.statsmodels.org/stable/release/version0.12.html>.
- [114] D. Alberg, M. Last, Short-term load forecasting in smart meters with sliding window-based ARIMA algorithms, *Vietnam J. Comput. Sci.* 5 (2018) 241–249. <https://doi.org/10.1007/s40595-018-0119-7>.
- [115] I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*, MIT Press, 2016.
- [116] S.S. Subbiah, J. Chinnappan, A review of short term load forecasting using deep learning, *Int. J. Emerg. Technol.* 11 (2020) 378–384. <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85084348260&partnerID=40&md5=edad5c98f6b932f71f111a990b24ef5a>.
- [117] N.U. Moroff, E. Kurt, J. Kamphues, Machine Learning and Statistics: A Study for assessing innovative Demand Forecasting Models, *Procedia Comput. Sci.* 180 (2021) 40–49. <https://doi.org/https://doi.org/10.1016/j.procs.2021.01.127>.
- [118] A. Gasparin, S. Lukovic, C. Alippi, Deep Learning for Time Series Forecasting: The Electric Load Case, *CoRR*. abs/1907.0 (2019). <http://arxiv.org/abs/1907.09207>.
- [119] L. Breiman, Random Forests, *Mach. Learn.* 45 (2001) 5–32. <https://doi.org/10.1023/A:1010933404324>.
- [120] S.M. Piryonesi, T.E. El-Diraby, Role of Data Analytics in Infrastructure Asset Management: Overcoming Data Size and Quality Problems, *J. Transp. Eng. Part B Pavements*. 146 (2020) 4020022. <https://doi.org/10.1061/JPEODX.0000175>.
- [121] G. Louppe, *Understanding Random Forests: From Theory to Practice*, (2015).
- [122] S. Misra, H. Li, Chapter 9 - Noninvasive fracture characterization based on the classification of sonic wave travel times, in: S. Misra, H. Li, J. He (Eds.), *Mach. Learn. Subsurf. Charact.*, Gulf Professional Publishing, 2020: pp. 243–287. <https://doi.org/https://doi.org/10.1016/B978-0-12-817736-5.00009-0>.
- [123] W. Koehrsen, An Implementation and Explanation of the Random Forest in Python, (n.d.). <https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76>.
- [124] D. Chicco, M.J. Warrens, G. Jurman, The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression

- analysis evaluation, PeerJ Comput. Sci. 7 (2021) e623. <https://doi.org/10.7717/peerj-cs.623>.
- [125] O.I. Abiodun, A. Jantan, A.E. Omolara, K.V. Dada, N.A. Mohamed, H. Arshad, State-of-the-art in artificial neural network applications: A survey, *Heliyon*. 4 (2018) e00938. <https://doi.org/https://doi.org/10.1016/j.heliyon.2018.e00938>.
- [126] L. Fausett, *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*, Prentice-Hall, Inc., USA, 1994.
- [127] K. Gurney, *An introduction to neural networks*, CRC press, 1997.
- [128] D. Kriesel, *A Brief Introduction to Neural Networks*, 2007. available.
- [129] M.A. Nielsen, *Neural Networks and Deep Learning*, Determination Press, 2015. <http://neuralnetworksanddeeplearning.com/>.
- [130] P. Baheti, 12 Types of Neural Network Activation Functions: How to Choose?, V7 Platf. (n.d.). <https://www.v7labs.com/blog/neural-networks-activation-functions>.
- [131] K. et al. Katanforoosh, Parameter optimization in neural networks, *DeepLearning.AI*. (2019). <https://www.deeplearning.ai/ai-notes/optimization/>.
- [132] A. Zhang, Z.C. Lipton, M. Li, A.J. Smola, *Dive into Deep Learning*, 2020.
- [133] H. Li, Z. Xu, G. Taylor, C. Studer, T. Goldstein, Visualizing the Loss Landscape of Neural Nets, (2018).
- [134] S. Ruder, An overview of gradient descent optimization algorithms, (2017).
- [135] N. Qian, On the momentum term in gradient descent learning algorithms, *Neural Networks*. 12 (1999) 145–151. [https://doi.org/https://doi.org/10.1016/S0893-6080\(98\)00116-6](https://doi.org/https://doi.org/10.1016/S0893-6080(98)00116-6).
- [136] Y. Nesterov, A method for unconstrained convex minimization problem with the rate of convergence $o(1/k^2)$, in: 1983.
- [137] Ilya Sutskever, TRAINING RECURRENT NEURAL NETWORKS, Graduate Department of Computer Science University of Toronto, Toronto, 2013.
- [138] J. Duchi, E. Hazan, Y. Singer, Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, *J. Mach. Learn. Res.* 12 (2011) 2121–2159. <http://jmlr.org/papers/v12/duchi11a.html>.
- [139] M.D. Zeiler, ADADELTA: An Adaptive Learning Rate Method, (2012).
- [140] K.S. Hinton, Geoffrey; Srivastava, Nitish Swersky, *Neural Networks for Machine Learning*, in: *Neural Networks Mach. Learn.*, Toronto, n.d.: pp. 26–30. http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.
- [141] D.P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, (2017).
- [142] T. Dozat, Incorporating Nesterov Momentum into Adam, in: 2016.
- [143] S.J. Reddi, S. Kale, S. Kumar, On the Convergence of Adam and Beyond, (2019).
- [144] E. Hoseinzade, S. Haratizadeh, CNNPred: CNN-based stock market prediction using several data sources, (2018).
- [145] J. Brownlee, *Deep Learning with Time Series Forecasting*, Machine Learning Mastery, n.d. <https://machinelearningmastery.com/deep-learning-for-time-series-forecasting/>.
- [146] I. Silva, D. Spatti, R.A. Flauzino, L. Bartocci Liboni, S. Reis Alves, Multilayer Perceptron Networks, in: 2017: pp. 55–115. https://doi.org/10.1007/978-3-319-43162-8_5.
- [147] M.A. Mercioni, S. Holban, The Most Used Activation Functions: Classic Versus Current, in: 2020 Int. Conf. Dev. Appl. Syst., 2020: pp. 141–145.

- <https://doi.org/10.1109/DAS49615.2020.9108942>.
- [148] J.N.C. Gonçalves, P. Cortez, M.S. Carvalho, N.M. Frazão, A multivariate approach for multi-step demand forecasting in assembly industries: Empirical evidence from an automotive supply chain, *Decis. Support Syst.* 142 (2021) 113452. <https://doi.org/https://doi.org/10.1016/j.dss.2020.113452>.
- [149] J.K. Mandal, A.K. Sinha, G. Parthasarathy, Application of recurrent neural network for short term load forecasting in electric power system, in: *Proc. ICNN'95 - Int. Conf. Neural Networks*, 1995: pp. 2694–2698 vol.5. <https://doi.org/10.1109/ICNN.1995.487837>.
- [150] H. Mori, T. Ogasawara, A recurrent neural network for short-term load forecasting, in: [1993] *Proc. Second Int. Forum Appl. Neural Networks to Power Syst.*, 1993: pp. 395–400. <https://doi.org/10.1109/ANN.1993.264315>.
- [151] P. Fisseha Berhane, Building your Recurrent Neural Network - Step by Step, (n.d.). https://datascience-enthusiast.com/DL/Building_a_Recurrent_Neural_Network-Step_by_Step_v1.html.
- [152] B. Dietrich, J. Walther, M. Weigold, E. Abele, Machine learning based very short term load forecasting of machine tools, *Appl. Energy.* 276 (2020). <https://doi.org/10.1016/j.apenergy.2020.115440>.
- [153] S. Hochreiter, J. Schmidhuber, Long Short-Term Memory, *Neural Comput.* 9 (1997) 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>.
- [154] R. Pascanu, T. Mikolov, Y. Bengio, On the difficulty of training Recurrent Neural Networks, (2013).
- [155] C. Cao, F. Liu, H. Tan, D. Song, W. Shu, W. Li, Y. Zhou, X. Bo, Z. Xie, Deep Learning and Its Applications in Biomedicine, Genomics, Proteomics & Bioinforma. 16 (2018) 17–32. <https://doi.org/https://doi.org/10.1016/j.gpb.2017.07.003>.
- [156] S. Muzaffar, A. Afshari, Short-Term Load Forecasts Using LSTM Networks, *Energy Procedia.* 158 (2019) 2922–2927. <https://doi.org/https://doi.org/10.1016/j.egypro.2019.01.952>.
- [157] W. Kong, Z.Y. Dong, Y. Jia, D.J. Hill, Y. Xu, Y. Zhang, Short-Term Residential Load Forecasting Based on LSTM Recurrent Neural Network, *IEEE Trans. Smart Grid.* 10 (2019) 841–851. <https://doi.org/10.1109/TSG.2017.2753802>.
- [158] R. Jiao, T. Zhang, Y. Jiang, H. He, Short-Term Non-Residential Load Forecasting Based on Multiple Sequences LSTM Recurrent Neural Network, *IEEE Access.* 6 (2018) 59438–59448. <https://doi.org/10.1109/ACCESS.2018.2873712>.
- [159] D.-C. Wu, B. Bahrami Asl, A. Razban, J. Chen, Air compressor load forecasting using artificial neural network, *Expert Syst. Appl.* 168 (2021) 114209. <https://doi.org/https://doi.org/10.1016/j.eswa.2020.114209>.
- [160] Jian Zheng, Cencen Xu, Ziang Zhang, Xiaohua Li, Electric load forecasting in smart grids using Long-Short-Term-Memory based Recurrent Neural Network, in: 2017 51st Annu. Conf. Inf. Sci. Syst., 2017: pp. 1–6. <https://doi.org/10.1109/CISS.2017.7926112>.
- [161] K.M. Powell, A. Sriprasad, W.J. Cole, T.F. Edgar, Heating, cooling, and electrical load forecasting for a large-scale district energy system, *Energy.* 74 (2014) 877–885. <https://doi.org/https://doi.org/10.1016/j.energy.2014.07.064>.
- [162] H.D. Nguyen, K.P. Tran, S. Thomassey, M. Hamad, Forecasting and Anomaly Detection approaches using LSTM and LSTM Autoencoder techniques with the applications in supply chain management, *Int. J. Inf. Manage.* 57 (2021) 102282.

- <https://doi.org/https://doi.org/10.1016/j.ijinfomgt.2020.102282>.
- [163] C. Olah, Understanding LSTM networks., (n.d). <http://colah.github.io/posts/2015-08-Understanding-LSTMs>.
- [164] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, (2014).
- [165] I. Sutskever, O. Vinyals, Q. V Le, Sequence to Sequence Learning with Neural Networks, CoRR. abs/1409.3 (2014). <http://arxiv.org/abs/1409.3215>.
- [166] O. Irsoy, C. Cardie, Deep Recursive Neural Networks for Compositionality in Language, in: Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, K.Q. Weinberger (Eds.), Adv. Neural Inf. Process. Syst., Curran Associates, Inc., 2014. <https://proceedings.neurips.cc/paper/2014/file/2cfd4560539f887a5e420412b370b361-Paper.pdf>.
- [167] S. Ungureanu, V. Topa, A.C. Cziker, Analysis for Non-Residential Short-Term Load Forecasting Using Machine Learning and Statistical Methods with Financial Impact on the Power Market, Energies. 14 (2021). <https://doi.org/10.3390/en14216966>.
- [168] K. Lu, X.R. Meng, W.X. Sun, R.G. Zhang, Y.K. Han, S. Gao, D. Su, {GRU}-based Encoder-Decoder for Short-term {CHP} Heat Load Forecast, 392 (2018) 62173. <https://doi.org/10.1088/1757-899x/392/6/062173>.
- [169] G.P.H. Styan, Hadamard products and multivariate statistical analysis, Linear Algebra Appl. 6 (1973) 217–240. [https://doi.org/https://doi.org/10.1016/0024-3795\(73\)90023-2](https://doi.org/https://doi.org/10.1016/0024-3795(73)90023-2).
- [170] S. Ungureanu, V. Topa, A.C. Cziker, Deep Learning for Short-Term Load Forecasting—Industrial Consumer Case Study, Appl. Sci. 11 (2021). <https://doi.org/10.3390/app112110126>.
- [171] X. Li, W. Zhuang, H. Zhang, Short-term Power Load Forecasting Based on Gate Recurrent Unit Network and Cloud Computing Platform, Proc. 4th Int. Conf. Comput. Sci. Appl. Eng. (2020).
- [172] B. Lim, S. Zohren, Time Series Forecasting With Deep Learning: A Survey, (2020).
- [173] R. Chandra, S. Goyal, R. Gupta, Evaluation of Deep Learning Models for Multi-Step Ahead Time Series Prediction, IEEE Access. 9 (2021) 83105–83123. <https://doi.org/10.1109/ACCESS.2021.3085085>.
- [174] M. Alhussein, K. Aurangzeb, S.I. Haider, Hybrid CNN-LSTM Model for Short-Term Individual Household Load Forecasting, IEEE Access. 8 (2020) 180544–180557. <https://doi.org/10.1109/ACCESS.2020.3028281>.
- [175] J. Donahue, L.A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, T. Darrell, Long-term Recurrent Convolutional Networks for Visual Recognition and Description, (2016).
- [176] R. Pascanu, C. Gulcehre, K. Cho, Y. Bengio, How to Construct Deep Recurrent Neural Networks, (2014).
- [177] T.-Y. Kim, S.-B. Cho, Predicting residential energy consumption using CNN-LSTM neural networks, Energy. 182 (2019) 72–81. <https://doi.org/https://doi.org/10.1016/j.energy.2019.05.230>.
- [178] R. Yan, J. Liao, J. Yang, W. Sun, M. Nong, F. Li, Multi-hour and multi-site air quality index forecasting in Beijing using CNN, LSTM, CNN-LSTM, and spatiotemporal clustering, Expert Syst. Appl. 169 (2021) 114513.

- <https://doi.org/https://doi.org/10.1016/j.eswa.2020.114513>.
- [179] S.H. Rafi, Nahid-Al-Masood, S.R. Deeba, E. Hossain, A Short-Term Load Forecasting Method Using Integrated CNN and LSTM Network, *IEEE Access*. 9 (2021) 32436–32448. <https://doi.org/10.1109/ACCESS.2021.3060654>.
 - [180] R.J. Hyndman, A.B. Koehler, Another look at measures of forecast accuracy, *Int. J. Forecast.* 22 (2006) 679–688. <https://doi.org/https://doi.org/10.1016/j.ijforecast.2006.03.001>.
 - [181] E. Vivas, H. Allende-Cid, R. Salas, A Systematic Review of Statistical and Machine Learning Methods for Electrical Power Forecasting with Reported MAPE Score, *Entropy*. 22 (2020). <https://doi.org/10.3390/e22121412>.
 - [182] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, A. Müller, J. Nothman, G. Louppe, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, É. Duchesnay, *Scikit-learn: Machine Learning in Python*, (2018).
 - [183] Q. Duan, J. Liu, D. Zhao, Short term electric load forecasting using an automated system of model choice, *Int. J. Electr. Power Energy Syst.* 91 (2017) 92–100. <https://doi.org/https://doi.org/10.1016/j.ijepes.2017.03.006>.
 - [184] P. dr. ing. M. Gavrilas, INTRODUCERE ÎN PIAȚA DE ENERGIE ELECTRICĂ, (n.d.). <http://www.mihai-gavrilas.ieeia.tuiasi.ro/>.
 - [185] M. Nakamura, T. Nakashima, T. Niimura, Electricity markets volatility: estimates, regularities and risk management applications, *Energy Policy*. 34 (2006) 1736–1749. <https://doi.org/https://doi.org/10.1016/j.enpol.2004.12.019>.
 - [186] Radu Bindiu, PROGNOZA PE TERMEN SCURT A CONSUMULUI DE ENERGIE ELECTRICĂ, 2011.
 - [187] J.T. Reilly, From microgrids to aggregators of distributed energy resources. The microgrid controller and distributed energy management systems, *Electr. J.* 32 (2019) 30–34. <https://doi.org/https://doi.org/10.1016/j.tej.2019.05.007>.

LISTA FIGURILOR

Figura 1.1 Integrarea consumatorului industrial în rețele electrice inteligente	12
Figura 1.2 Stadiul de finanțare al SG în UE la nivelul anului 2015.....	13
Figura 1.3 Integrarea consumatorului industrial în rețele inteligente.....	17
Figura 1.4 Diagramă bloc pentru controlul curbei de sarcină	21
Figura 1.5 Integrarea prognozei în contextul SG	25
Figura 1.6 Consumul final de energie electrică în industrie (2019) ⁵	28
Figura 1.7 Ponderea consumului de energie în totalul de energie – 2016 ⁵	28
Figura 1.8 Ponderea consumului de electricitate în industrie – 2016 ⁵	28
Figura 1.9 Impactul consumului de energie asupra emisiilor de CO ₂	29
Figura 2.1 Aplicabilitatea prognozei de consum de electricitate[26]	31
Figura 2.2 Tipuri de prognoza : calitative și cantitative	32
Figura 2.3 Legătura dintre volumul de date și performanță[42]	37
Figura 2.4 Domeniile inteligenței artificiale ⁷	37
Figura 2.5 Clasificarea algoritmilor de învățare automată[43].....	37
Figura 3.1 Dezechilibrul în SEN în ianuarie 2021	43
Figura 4.1 Curbă de sarcină pentru fabrica de parchet (FP)	47
Figura 4.2 Curbă de sarcină pentru fabrica de producție de mobilier (FF).....	48
Figura 4.3 Curbă de sarcină supermarket (SM)	48
Figura 4.4 Curbă de sarcină pentru auto showroom și service (AS)	48
Figura 4.5 Curbă de sarcină abator (FPF)	49
Figura 4.6 Curbă de sarcină – profil săptămânal.....	49
Figura 4.7 Cluster de consumatori analizați.....	50
Figura 4.8 Sumarizarea procesului de alegere a metodelor de prognoză.....	51
Figura 4.9 Consumatorul industrial – fabrică de parchet	54
Figura 4.10 Diagrama Sankey – fabrica de parchet (FP)	55
Figura 4.11 Consumatorul industrial – fabrică de mobilă.....	56
Figura 4.12 Consumatorul comercial – supermarket (bilanț real anual).....	56
Figura 4.13 Curbele de sarcină agregate ale consumatorilor	57
Figura 4.14 Variabile exogene și autoregresive.....	59
Figura 4.15 Interesul de căutare pentru platforme ML	61
Figura 5.1 Metodologia generală a lucrării	63
Figura 5.2 Distribuția seriilor de timp.....	64
Figura 5.3 Funcția de autocorelație aplicată pe consumul agregat.....	67
Figura 5.4 Funcția de autocorelație parțială aplicată pe consumul agregat	68
Figura 5.5 Principiul perioadei glisante [117].....	70
Figura 5.6 Algoritmii RF pentru prognoza[121]	74
Figura 5.7 RF cu doi arbori decizionali	75
Figura 5.8 Clasificarea rețelelor neuronale artificiale [125]	76
Figura 5.9 Structura unei rețele neuronale artificiale [127].....	78
Figura 5.10 Algoritmii gradient descendent în funcție de un parametru	83
Figura 5.11 Funcția cost în funcție de doi parametri (w,b)[131]	83
Figura 5.12 Metodologia completă de implementare a prognozei.....	89
Figura 5.13 Pseudocod folosit pentru implementarea software	91
Figura 5.14 Structura perceptron multi-strat.....	92
Figura 5.15 Învățarea supravegheată cu MLP	92
Figura 5.16 Neuron folosit în rețeaua recurentă neuronală [150]	94
Figura 5.17 Rețele neuronale recurente – vizualizare desfășurată.....	95
Figura 5.18 Modelul folosit pentru testare (prognoza efectivă).....	96
Figura 5.19 Celulă LSTM . Adaptată din [162].....	98
Figura 5.20 Rețea neuronală recurentă cu mai multe straturi ascunse[166]	100

Figura 5.21 Celula GRU Gate Recurrent Unit. Adaptat din [162].....	101
Figura 5.22 Implementarea GRU pentru prognoza curbele de sarcină [169].....	104
Figura 5.23 Rețea LSTM “encoder-decoder” [163]	106
Figura 5.24 Rețea neuronală combinată CNN - LSTM [176].....	107
Figura 6.1 Pseudo-cod pentru implementarea algoritmilor ML.....	115
Figura 6.2 Prognoză (h=24) cu metoda naivă I (MAPE = 7,64%)	118
Figura 6.3 Prognoză (h=24) cu metoda naivă II (MAPE = 9,72%).....	118
Figura 6.4 Prognoză (h=24) cu NE (MAPE = 9,75%).....	119
Figura 6.5 Prognoză cu metoda AR (MAPE = 6,31%)	120
Figura 6.6 Prognoză cu metoda MA (MAPE = 8,24%)	120
Figura 6.7 Prognoză realizată cu SARIMA (MAPE = 7,56%).....	121
Figura 6.8 Prognoză realizată cu MLP (MAPE = 5,95%).....	121
Figura 6.9 Prognoză realizată cu RNN (MAPE = 6,61%)	122
Figura 6.10 Prognoză realizată cuLSTM (MAPE = 5,67%)	122
Figura 6.11 Prognoză realizată cu LSTM encoder-decoder (MAPE = 6,23%).....	123
Figura 6.12 Prognoză realizată cu GRU (MAPE = 5,28%)	123
Figura 6.13 Prognoză realizată cu CNN-LSTM (MAPE = 6,97%).....	124
Figura 6.14 Consumului real vs prognozat - metode tradiționale.....	125
Figura 6.15 Consumului real vs prognozat - machine learning.....	125
Figura 6.16 Consumului real vs prognozat - LSTM.....	126
Figura 6.17 Consumului real vs prognozat - GRU.....	126
Figura 6.18 Consumului real vs prognozat - GRU și LSTM.....	127
Figura 6.19 Evoluția zilnică a prognozei (GRU 24 3 100 100 48 24)	131
Figura 6.20 Evidențierea diversilor algoritmi GRU antrenați diferit	132
Figura 6.21 Erori obținute cu GRU 24 3 100 100 48 24.....	133
Figura 6.22 Evoluția erorilor în funcție de DL_{index}	135
Figura 6.23 Evoluția erorilor în funcție de numărul de parametri antrenați	136
Figura 6.24 Rezultatele arhitecturii GRU cu 5.336.964,00 parametri.....	137
Figura 7.1 Influența temperaturii în consumul de electricitate.....	140
Figura 7.2 Influența temperaturii în consumul lunar al supermarket-ului.....	140
Figura 7.3 Consum real vs prognozat - GRU, GRU+LSTM și AR.....	145
Figura 7.4 Consum real vs prognozat - AR, LSTM, MLP și RNN.....	146
Figura 7.5 Consum real vs prognozat - GRU.....	147
Figura 7.6 Consum real vs prognozat - LSTM-GRU și LSTM.....	147
Figura 7.7 MAPE zilnic - GRU și LSTM pentru consumatorul FF	148
Figura 7.8 Prognoze tradiționale individuale	149
Figura 7.9 Prognoze cu ML individuale.....	150
Figura 7.10 Consum de electricitate fabrică producție parchet (PF)	151
Figura 7.11 Rezultate comparative pentru diferitele metode aplicate	152
Figura 8.1 Prețuri orare euro/MWh pentru PZU și PE	156
Figura 8.2 Comparația MAPE cu costul erorilor pentru prognoza pe 24 ore.....	158
Figura 8.3 Comparația MAPE cu costul erorilor pentru prognoza pe 48 ore.....	159
Figura 8.4 Comparația MAPE cu costul erorilor pentru prognoza pe 168 ore.....	160
Figura 8.5 Evaluarea zilnică a erorilor MAPE obținute cu GRU (24 ore).....	162
Figura 8.6 Evaluarea zilnică a erorilor MAPE obținute cu LSTM (24 ore).....	163

LISTA TABELELOR

Tabel 1.1 Impactul SG pentru piața de energie electrică.....	18
Tabel 1.2 Cerințe tehnice	20
Tabel 1.3 Pași pentru implementare DR.....	20
Tabel 1.4 Exemplu pentru identificarea scenariilor	22
Tabel 1.5 Consumul final de energie electrică pe activități industriale în anul 2019	27
Tabel 2.1 Tipuri de prognoză pentru de consum de electricitate	30
Tabel 2.2 Evidențierea studiilor analizate în literatură.....	38
Tabel 3.1 Importanța prognozei de consum de electricitate.....	42
Tabel 4.1 Date reprezentative pentru cluster-ul de consumatori.....	46
Tabel 4.2 Instalare centrală de măsură pentru obținerea datelor	52
Tabel 4.3 Panou de control și interfața pentru accesarea datelor	52
Tabel 4.4 Variabile exogene meteorologice luate în calcul pentru prognoză.....	59
Tabel 4.5 Valorile meteorologice orare folosite pentru prognoză.....	60
Tabel 4.6 Platforme pentru învățare aprofundată.....	62
Tabel 5.1 Analiză statistică Dickey-Fuller.....	64
Tabel 5.2 Analiză statistică. $R^2= 0,778$	66
Tabel 5.3 Criterii de decizie pentru RF.....	73
Tabel 5.4 Configurarea RF.....	75
Tabel 5.5 Funcții de activare	79
Tabel 5.6 Exemplu de reprezentare a datelor.....	84
Tabel 5.7 Variații ale algoritmului gradient descendent	86
Tabel 5.8 Parametrii modelului LSTM	98
Tabel 5.9 Argumentele neuronilor GRU și Dense (folosit în stratul de ieșire)	100
Tabel 5.10 Argumentele neuronilor GRU și Dense (folosit în stratul de ieșire)	105
Tabel 5.11 Formule folosite pentru evaluare prognoză	108
Tabel 5.12 Criteriu calitativ pentru MAPE	109
Tabel 6.1 Parametrii folosiți în construirea metodelor de prognoză.....	113
Tabel 6.2 Rezultate ($h=24$) pentru prognoză agregată cu metode tradiționale.....	117
Tabel 6.3 Rezultate ($h=24$) pentru prognoză agregată cu ML	117
Tabel 6.4 Configurarea rețelelor neuronale în funcție de orizontul de timp prognozat.....	128
Tabel 6.5 Măsurători prognoză pentru algoritmi în funcție de orizontul de prognoză.....	128
Tabel 6.6 Configurația rețelelor GRU folosite în studiu – exemplu codificare	130
Tabel 7.1 Parametrii considerați pentru evaluarea algoritmilor de prognoză individuală.....	141
Tabel 7.2 Rezultate prognoză cu metode tradiționale – consum individual.....	142
Tabel 7.3 Rezultate prognoză cu metode ML – consum individual.....	144
Tabel 7.4 Erori de prognoză pentru setul de date de testare pentru consumator FF	145
Tabel 8.1 Evaluarea prognozelor cu impactul financiar pe piața de energie – 24 ore	159
Tabel 8.2 Evaluarea prognozelor cu impactul financiar pe piața de energie – 48 ore	160
Tabel 8.3 Evaluarea prognozelor cu impactul financiar pe piața de energie – 168 ore	161

ANEXE

A1) Analiza Metodei GRU aplicată pe curba agregată pentru prognoza pe 24 ore- arhitecturi analizate pentru identificarea celor mai bune rezultate

Metodă	intrare	strataturi ascunse	iesire	timp	Epoci	param,	cost	optim,	MAPE	ME	MAE	MPE	RMSE
GRU	24	3	24	100	10	120876	mse	adam	antrenare				
				100					0,088	0,2138	0,3256	0,0657	0,4586
				48					testare				
									0,0707	0,0875	0,3207	0,0255	0,4189
GRU	24	3	24	100	30	120876	mse	adam	antrenare				
				100					0,0482	0,0417	0,1871	0,0151	0,2626
				48					testare				
									0,0563	-0,0244	0,2602	-0,0004	0,3636
GRU	24	3	24	100	50	120876	mse	adam	antrenare				
				100					0,0434	0,0909	0,172	0,0234	0,2467
				48					testare				
									0,0542	-0,0216	0,2497	-0,0035	0,3364
GRU	24	3	24	100	60	120876	mse	adam	antrenare				
				100					0,0337	0,0177	0,137	0,0044	0,1951
				48					testare				
									0,0593	-0,0976	0,2761	-0,0191	0,3882
GRU	24	3	24	100	100	120876	mse	adam	antrenare				
				100					0,03	-0,0783	0,1238	-0,0185	0,1575
				48					testare				
									0,0528	-0,007	0,2363	0,001	0,322
GRU	24	3	24	100	120	120876	mse	adam	antrenare				
				100					0,0243	0,0178	0,0993	0,0046	0,1313
				48					testare				
									0,0562	-0,0887	0,2602	-0,0161	0,3471
GRU	24	3	24	100	150	120876	mse	adam	antrenare				
				100					0,0335	0,015	0,1315	0,0042	0,1732
				48					testare				
									0,0588	-0,0912	0,2674	-0,019	0,3534
GRU	24	3	24	100	200	120876	mse	adam	antrenare				
				100					0,0303	-0,0088	0,1227	0,0013	0,1668
				48					testare				
									0,0641	-0,1365	0,2586	-0,0253	0,3425
GRU	24	4	24	24	10	59280	mse	adam	antrenare				
				48					0,0855	0,036	0,33	0,0203	0,4648
				72					testare				
									0,0786	-0,124	0,3632	-0,0228	0,5007
GRU	24	4	24	24	30	59280	mse	adam	antrenare				
				48					0,067	0,123	0,258	0,035	0,361
				72					testare				
									0,073	-0,075	0,339	-0,015	0,508
GRU	24	4	24	24	50	59280	mse	adam	antrenare				
				48					0,0397	-0,0112	0,1588	0,0039	0,2183
				72					testare				
									0,0534	-0,1089	0,2792	-0,0194	0,3663
				48									

MEtoda	intrare	straturii ascuse	iesire	timp	epoci	#param,	cost	optim,	MAPE	ME	MAE	MPE	RMSE
GRU	24	4	24	652,84	100	59280	mse	adam	antrenare				
									0,0262	0,0049	0,1063	0,0037	0,1411
									testare				
									0,0559	-0,1036	0,2917	-0,0194	0,4088
GRU	24	4	24	920,33	150	59280	mse	adam	antrenare				
									0,0465	-0,0063	0,1737	0,0082	0,2722
									testare				
									0,0606	-0,1032	0,2763	-0,014	0,3605
GRU	24	4	24	1277,18	200	59280	mse	adam	antrenare				
									0,0217	-0,019	0,0901	-0,0043	0,1168
									testare				
									0,0648	-0,1776	0,3027	-0,0379	0,4048
GRU	24	3	24	171,42	10	748536	mse	adam	antrenare				
									0,0631	0,0375	0,2431	0,0093	0,3445
									testare				
GRU	24	3	24	314,84	20	748536	mse	adam	antrenare				
									0,0407	-0,0359	0,1636	-0,0055	0,2292
									testare				
GRU	24	3	24	314,84	30	748536	mse	adam	antrenare				
									0,0331	-0,038	0,1333	-0,008	0,1948
									testare				
GRU	24	3	24	632,13	40	748536	mse	adam	antrenare				
									0,0304	-0,034	0,1236	-0,006	0,1664
									testare				
GRU	24	3	24	728,7	50	748536	mse	adam	antrenare				
									0,0302	-0,0082	0,0749	-0,0023	0,0965
									testare				
GRU	24	3	24	1444,19	100	748536	mse	adam	antrenare				
									0,0184	-0,0082	0,0749	-0,0023	0,0965
									testare				
GRU	24	3	24	2511,92	150	748536	mse	adam	antrenare				
									0,0161	0,0043	0,0648	0,0017	0,0827
									testare				
GRU	24	3	24	3535,24	200	748536	mse	adam	antrenare				
									0,0117	0,0007	0,0475	0,0012	0,0601
									testare				
GRU	24	3	24	59,59	10	83976	mse	adam	antrenare				
									0,0796	0,0387	0,3041	0,0248	0,4252
									testare				
									0,0698	-0,114	0,3251	-0,0175	0,4076

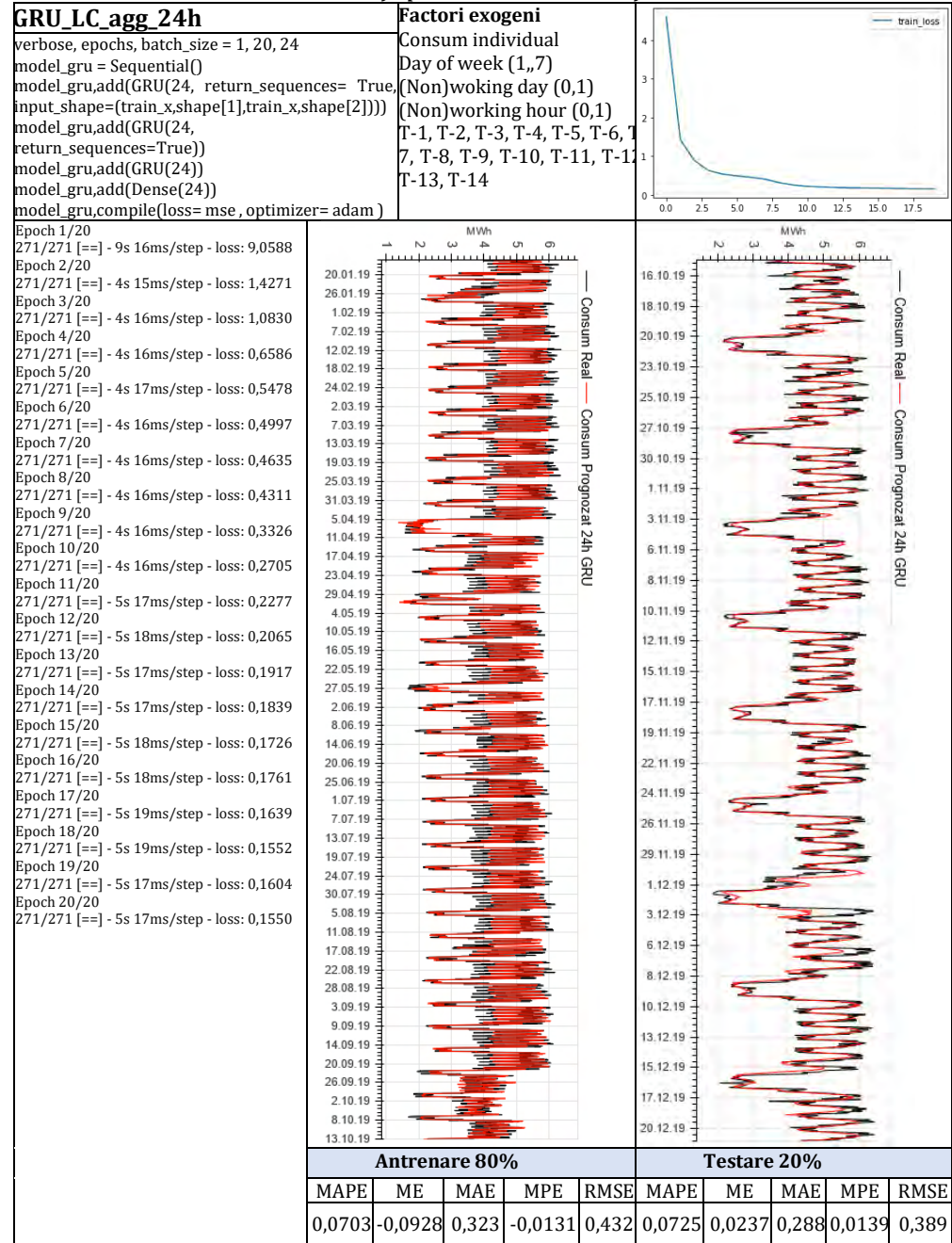
MEtoda	intrare	straturascunse	iesire	timp	epoci	#para,	cost	optim,	MAPE	ME	MAE	MPE	RMSE	
GRU	24	3	80	24	116,64	20	83976	mse	adam	antrenare				
			80							0,063	0,0126	0,2482	0,0092	0,35
			48							testare				
									0,065	-0,1119	0,303	-0,0217	0,4202	
GRU	24	3	80	24	167,48	30	83976	mse	adam	antrenare				
			80							0,0505	0,03	0,1988	0,0148	0,2724
			48							testare				
									0,0609	-0,081	0,2855	-0,0114	0,3711	
GRU	24	3	80	24	282,49	50	83976	mse	adam	antrenare				
			80							0,0375	0,0207	0,1473	0,0084	0,2107
			48							testare				
									0,0597	-0,0857	0,2749	-0,0139	0,3555	
GRU	24	3	80	24	613,24	100	83976	mse	adam	antrenare				
			80							0,0272	-0,0032	0,1105	0,001	0,1462
			48							testare				
									0,0556	-0,0965	0,2592	-0,0182	0,3482	
GRU	24	3	80	24	774,25	150	83976	mse	adam	antrenare				
			80							0,021	-0,0095	0,0815	-0,0023	0,1049
			48							testare				
									0,0587	-0,1128	0,2851	-0,0189	0,3833	
GRU	24	3	80	24	1099,81	200	83976	mse	adam	antrenare				
			80							0,0199	-0,0095	0,0815	-0,0023	0,1049
			48							testare				
									0,0699	-0,1128	0,2851	-0,0189	0,3833	
GRU	24	3	160	24	99	10	284,784	mse	adam	antrenare				
			160							0,072	0,0133	0,2808	0,0136	0,401
			60							testare				
									0,071	-0,1418	0,3286	-0,026	0,4242	
GRU	24	3	160	24	265,57	30	284,784	mse	adam	antrenare				
			160							0,0595	0,1784	0,2188	0,0508	0,3057
			60							testare				
									0,0622	0,0837	0,2701	0,0257	0,3544	
GRU	24	3	160	24	429,88	50	284,784	mse	adam	antrenare				
			160							0,0323	0,0212	0,13	0,007	0,179
			60							testare				
									0,0538	-0,0648	0,2465	-0,0108	0,3317	
GRU	24	3	160	24	854,64	100	284,784	mse	adam	antrenare				
			160							0,0277	0,0055	0,1102	0,0035	0,1554
			60							testare				
									0,0576	-0,0994	0,2678	-0,0185	0,3514	
GRU	24	3	160	24	1411,5	150	284,784	mse	adam	antrenare				
			160							0,0183	-0,0161	0,0744	-0,0039	0,0956
			60							testare				
									0,0642	-0,1344	0,2976	-0,0267	0,4065	
GRU	24	3	160	24	1752,9	200	284,784	mse	adam	antrenare				
			160							0,0143	-0,0067	0,0582	-0,001	0,0746
			60							testare				
									0,0691	-0,13	0,2819	-0,0239	0,3806	

MEtoda		intrare	strataturi ascunse	iesire	timp	epoci	#param,	cost	optim,	MAPE	ME	MAE	MPE	RMSE	
GRU		24	3	100	24	96,11	10	238236	mse	adam	antrenare				
				100							0,0606	-0,0946	0,2369	-0,018	0,3312
				168							testare				
										0,0699	-0,2287	0,3229	-0,0485	0,4065	
GRU		24	3	100	24	263,22	30	238236	mse	adam	antrenare				
				100							0,0473	-0,0376	0,1851	-0,0055	0,2627
				168							testare				
										0,0588	-0,1451	0,2777	-0,0285	0,3576	
GRU		24	3	100	24	486,86	50	238236	mse	adam	antrenare				
				100							0,0372	-0,0866	0,1576	-0,0172	0,2179
				168							testare				
										0,0683	-0,2143	0,3216	-0,043	0,4169	
GRU		24	3	100	24	872,35	100	238236	mse	adam	antrenare				
				100							0,0254	-0,0362	0,1022	-0,0097	0,1307
				168							testare				
										0,065	-0,1518	0,3042	-0,028	0,4002	
GRU		24	3	100	24	1215,66	150	238236	mse	adam	antrenare				
				100							0,0173	-0,0076	0,0699	-0,0016	0,0896
				168							testare				
										0,06	-0,131	0,2774	-0,0228	0,3728	
GRU		24	3	100	24	1870,81	200	238236	mse	adam	antrenare				
				100							0,0152	-0,0012	0,0633	0,0006	0,0818
				168							testare				
										0,0612	-0,142	0,2894	-0,0268	0,3994	
GRU		24	5	100	24	128,48	10	282324	mse	adam	antrenare				
				100							0,0744	0,0839	0,2879	0,0309	0,411
				100							testare				
										0,0664	-0,0731	0,3051	-0,0115	0,3868	
GRU		24	5	100	24	376,13	30	282324	mse	adam	antrenare				
				100							0,0369	-0,0519	0,1493	-0,0104	0,213
				100							testare				
										0,0626	-0,1624	0,2941	-0,0319	0,3939	
GRU		24	5	100	24	624,95	50	282324	mse	adam	antrenare				
				100							0,03	0,0183	0,118	0,0075	0,1612
				100							testare				
										0,0559	-0,0756	0,2587	-0,0131	0,346	
GRU		24	5	100	24	1245,31	100	282324	mse	adam	antrenare				
				100							0,0313	-0,0206	0,1215	-0,0039	0,1697
				100							testare				
										0,0618	-0,1156	0,2855	-0,0253	0,3685	
GRU		24	5	100	24	1621,66	150	282324	mse	adam	antrenare				
				100							0,0169	0,0135	0,0689	0,0042	0,0879
				100							testare				
										0,0569	-0,0646	0,2594	-0,0138	0,36	

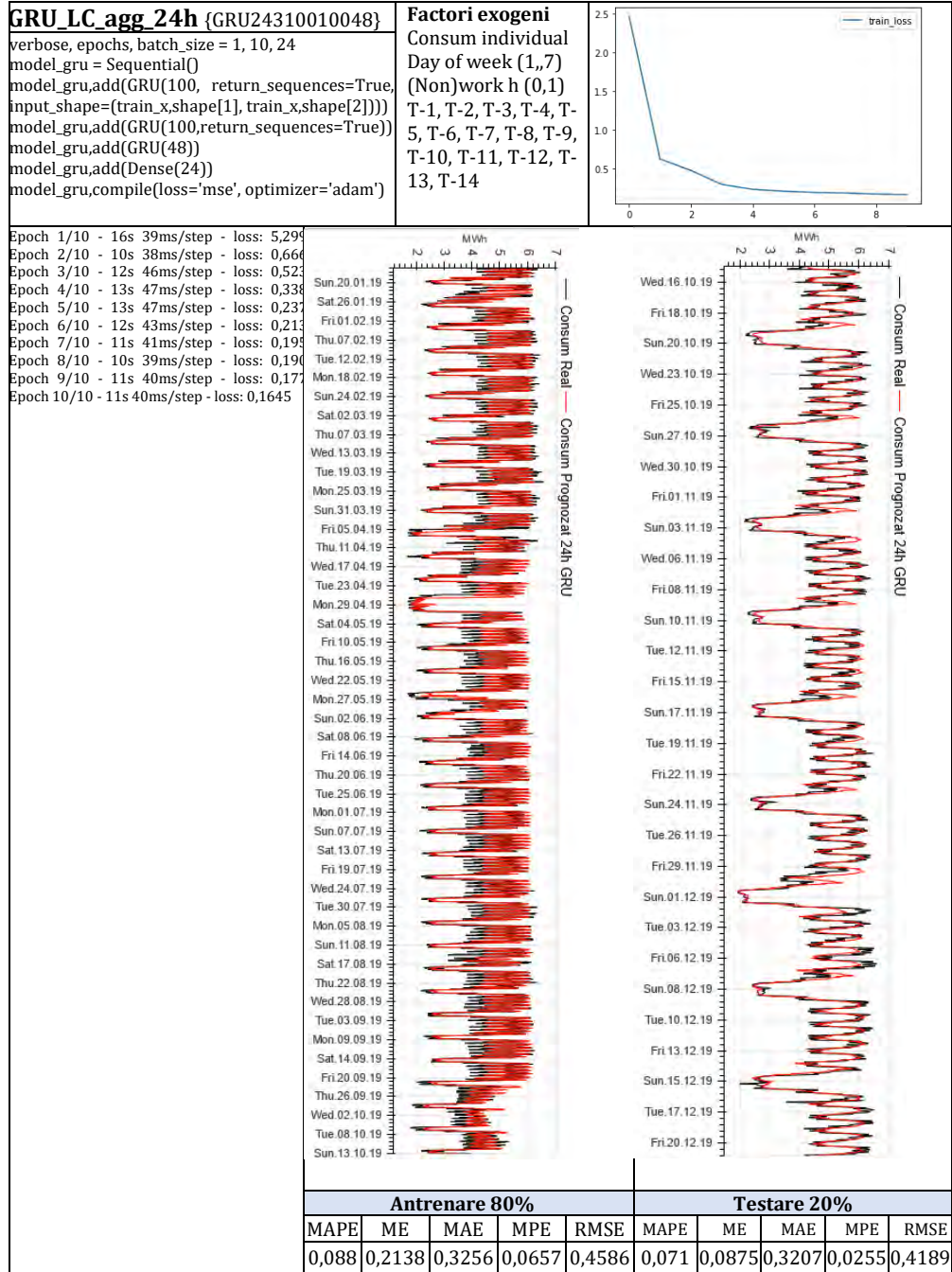
MEtoda	intrare	straturascunse	iesire	timp	epoci	#param,	cost	optim,	MAPE	ME	MAE	MPE	RMSE
GRU	24	5	24	100	200	282324	mse	adam	antrenare				
				100					0,0156	0,0119	0,0634	0,0026	0,0814
				100					testare				
				100					0,0598	-0,0676	0,2785	-0,0125	0,3713
				100									
GRU	24	5	24	200	10	1,104,624	mse	adam	antrenare				
				200					0,0693	-0,1085	0,2778	-0,0176	0,3861
				200					testare				
				200					0,0763	-0,2418	0,3518	-0,0514	0,4464
				200									
GRU	24	5	24	200	30	1,104,624	mse	adam	antrenare				
				200					0,0449	0,1425	0,1832	0,034	0,249
				200					testare				
				200					0,0566	0,0909	0,2601	0,0203	0,352
				200									
GRU	24	5	24	200	50	1,104,624	mse	adam	antrenare				
				200					0,0365	-0,0278	0,1413	-0,0051	0,2223
				200					testare				
				200					0,0618	-0,1298	0,284	-0,0271	0,3745
				200									
GRU	24	5	24	200	100	1,104,624	mse	adam	antrenare				
				200					0,0236	-0,0025	0,0961	0,0001	0,1277
				200					testare				
				200					0,0566	-0,0817	0,266	-0,0176	0,3612
				200									
GRU	24	5	24	200	150	1,104,624	mse	adam	antrenare				
				200					0,0147	-0,0072	0,0596	-0,0013	0,0763
				200					testare				
				200					0,0581	-0,1142	0,2752	-0,0224	0,3764
				200									
GRU	24	5	24	200	200	1,104,624	mse	adam	antrenare				
				200					0,018	-0,0036	0,0736	-0,0007	0,0946
				200					testare				
				200					0,0588	-0,0752	0,2668	-0,0152	0,3519
				200									
GRU	24	1	200	24	10	139824	mse	adam	antrenare				
									0,0655	0,0157	0,2448	0,0133	0,3606
									0,0629	-0,1102	0,2861	-0,0203	0,3686
GRU	24	1	200	24	30	139824	mse	adam	antrenare				
									0,0469	-0,0369	0,1827	-0,0073	0,2791
									0,0621	-0,1593	0,287	-0,0344	0,3802
GRU	24	1	200	24	50	139824	mse	adam	antrenare				
									0,0347	0,0062	0,1401	0,0031	0,2057
									0,0559	-0,0791	0,2606	-0,0141	0,3444

MEtoda	intrare		straturi ascunse	iesire	timp	epoci	#param,	cost	optim,	MAPE	ME	MAE	MPE	RMSE
GRU	24	1	200	24	523,29	100	139824	mse	adam	antrenare				
										0,0329	0,0073	0,1317	0,0018	0,1874
										testare				
										0,0532	-0,0828	0,2484	-0,0132	0,3307
GRU	24	1	200	24	805,74	150	139824	mse	adam	antrenare				
										0,0346	0,0528	0,1389	0,0145	0,1827
										testare				
										0,0533	-0,0303	0,2483	-0,0013	0,3318
GRU	24	1	200	24	862,03	200	139824	mse	adam	antrenare				
										0,0254	0,0538	0,1038	0,014	0,1352
										testare				
										0,0578	-0,0429	0,2701	-0,0063	0,3609

A2) Analiza Metodei GRU aplicată pe curba agregată pentru prognoza pe 24 ore- vizualizare rezultate și parametrii utilizați



GRU_LC_agg_24h {GRU24310010048}	Factori exogeni					train_loss				
verbose, epochs, batch_size = 1, 100, 24 model_gru = Sequential() model_gru.add(GRU(100, return_sequences= True, input_shape=(train_x_shape[1], train_x_shape[2]))) model_gru.add(GRU(100,return_sequences=True)) model_gru.add(GRU(48)) model_gru.add(Dense(24)) model_gru.compile(loss='mse',optimizer='adam')	Consum individual Day of week (1,,7) (Non)working hour (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14									
Epoch 1/100- 10s 20ms/step - loss: 5,7237 Epoch 2/100- 6s 22ms/step - loss: 0,7592 Epoch 3/100- 6s 22ms/step - loss: 0,5607 Epoch 4/100- 6s 22ms/step - loss: 0,3808 Epoch 5/100- 6s 22ms/step - loss: 0,2688 Epoch 6/100- 6s 22ms/step - loss: 0,2276 Epoch 7/100- 6s 22ms/step - loss: 0,1950 Epoch 8/100- 6s 22ms/step - loss: 0,1924 Epoch 9/100- 6s 22ms/step - loss: 0,1826 Epoch 10/100- 6s 22ms/step - loss: 0,1743 Epoch 11/100- 6s 22ms/step - loss: 0,1822 Epoch 12/100- 6s 22ms/step - loss: 0,1583 Epoch 13/100- 6s 22ms/step - loss: 0,1580 Epoch 14/100- 6s 22ms/step - loss: 0,1392 Epoch 15/100- 6s 22ms/step - loss: 0,1288 Epoch 16/100- 6s 22ms/step - loss: 0,1437 Epoch 17/100- 6s 22ms/step - loss: 0,1335 Epoch 18/100- 6s 22ms/step - loss: 0,1070 Epoch 19/100- 6s 22ms/step - loss: 0,1015 Epoch 20/100- 6s 22ms/step - loss: 0,0979 Epoch 21/100- 6s 22ms/step - loss: 0,0959 Epoch 22/100- 6s 22ms/step - loss: 0,0975 Epoch 23/100- 6s 22ms/step - loss: 0,0734 Epoch 24/100- 6s 22ms/step - loss: 0,0753 Epoch 25/100- 6s 22ms/step - loss: 0,0681 Epoch 26/100- 6s 22ms/step - loss: 0,0765 Epoch 27/100- 6s 22ms/step - loss: 0,0750 Epoch 28/100- 6s 22ms/step - loss: 0,1026 Epoch 29/100- 6s 22ms/step - loss: 0,0778 Epoch 30/100- 6s 22ms/step - loss: 0,0792 Epoch 31/100- 6s 22ms/step - loss: 0,1055 Epoch 32/100- 6s 22ms/step - loss: 0,0727 Epoch 33/100- 6s 22ms/step - loss: 0,0605 Epoch 34/100- 6s 22ms/step - loss: 0,0639 Epoch 35/100- 6s 22ms/step - loss: 0,0518 Epoch 36/100- 6s 22ms/step - loss: 0,0653 Epoch 37/100- 6s 22ms/step - loss: 0,0516 Epoch 38/100- 6s 22ms/step - loss: 0,0514 Epoch 39/100- 6s 22ms/step - loss: 0,0525 Epoch 40/100- 6s 22ms/step - loss: 0,0500 Epoch 41/100- 6s 22ms/step - loss: 0,0481 Epoch 42/100- 6s 22ms/step - loss: 0,0676 Epoch 43/100- 6s 23ms/step - loss: 0,0536 Epoch 44/100- 6s 22ms/step - loss: 0,1185 Epoch 45/100- 6s 22ms/step - loss: 0,0615 Epoch 46/100- 6s 22ms/step - loss: 0,0482 Epoch 47/100- 6s 22ms/step - loss: 0,0464 Epoch 48/100- 6s 22ms/step - loss: 0,0452 Epoch 49/100- 6s 22ms/step - loss: 0,0437 Epoch 50/100- 6s 22ms/step - loss: 0,0461 Epoch 51/100- 6s 22ms/step - loss: 0,0404 Epoch 52/100- 6s 22ms/step - loss: 0,0416 Epoch 53/100- 6s 22ms/step - loss: 0,0400 Epoch 54/100- 6s 22ms/step - loss: 0,0399 Epoch 55/100- 6s 22ms/step - loss: 0,0443 Epoch 56/100- 6s 22ms/step - loss: 0,0448 Epoch 57/100- 6s 22ms/step - loss: 0,0444 Epoch 58/100- 6s 22ms/step - loss: 0,0383 Epoch 59/100- 6s 22ms/step - loss: 0,0412 Epoch 60/100- 6s 22ms/step - loss: 0,0373 Epoch 87/100- 6s 22ms/step - loss: 0,0235 Epoch 99/100- 6s 22ms/step - loss: 0,0271 Epoch 100/100- 6s 22ms/step - loss: 0,0183										
Antrenare 80%					Testare 20%					
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE	
0,0242	0,0136	0,0994	0,0037	0,131	0,0623	-0,1126	0,290	-0,023	0,414	



GRU_LC_agg_24h {GRU24310010048}

```

verbose, epochs, batch_size = 1, 30, 24
model_gru = Sequential()
model_gru.add(GRU(100, return_sequences=True,
input_shape=(train_x.shape[1], train_x.shape[2])))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(48))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')

```

Factori exogeni
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14

```

Epoch 1/30 - 16s 41ms/step - loss: 5,59
Epoch 2/30 - 11s 42ms/step - loss: 0,80
Epoch 3/30 - 12s 44ms/step - loss: 0,53
Epoch 4/30 - 11s 42ms/step - loss: 0,38
Epoch 5/30 - 12s 42ms/step - loss: 0,23
Epoch 6/30 - 11s 41ms/step - loss: 0,21
Epoch 7/30 - 11s 40ms/step - loss: 0,18
Epoch 8/30 - 11s 41ms/step - loss: 0,19
Epoch 9/30 - 11s 40ms/step - loss: 0,16
Epoch 10/30 - 11s 42ms/step - loss: 0,18
Epoch 11/30 - 11s 41ms/step - loss: 0,15
Epoch 12/30 - 11s 41ms/step - loss: 0,15
Epoch 13/30 - 12s 43ms/step - loss: 0,15
Epoch 14/30 - 11s 41ms/step - loss: 0,12
Epoch 15/30 - 12s 42ms/step - loss: 0,12
Epoch 16/30 - 11s 41ms/step - loss: 0,12
Epoch 17/30 - 12s 43ms/step - loss: 0,12
Epoch 18/30 - 12s 44ms/step - loss: 0,12
Epoch 19/30 - 11s 41ms/step - loss: 0,10
Epoch 20/30 - 10s 38ms/step - loss: 0,09
Epoch 21/30 - 10s 38ms/step - loss: 0,12
Epoch 22/30 - 12s 44ms/step - loss: 0,08
Epoch 23/30 - 13s 48ms/step - loss: 0,07
Epoch 24/30 - 12s 44ms/step - loss: 0,07
Epoch 25/30 - 11s 42ms/step - loss: 0,07
Epoch 26/30 - 12s 43ms/step - loss: 0,07
Epoch 27/30 - 11s 42ms/step - loss: 0,08
Epoch 28/30 - 12s 44ms/step - loss: 0,06
Epoch 29/30 - 11s 41ms/step - loss: 0,06
Epoch 30/30 - 11s 40ms/step - loss: 0,0572

```

Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,048	0,0417	0,1871	0,0151	0,2626	0,056	-0,0244	0,2602	0,0004	0,3636

GRU_LC_agg_24h {GRU24310010048}

```

verbose, epochs, batch_size = 1, 50, 24
model_gru = Sequential()
model_gru.add(GRU(100, return_sequences=True,
input_shape=(train_x.shape[1], train_x.shape[2])))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(48))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')

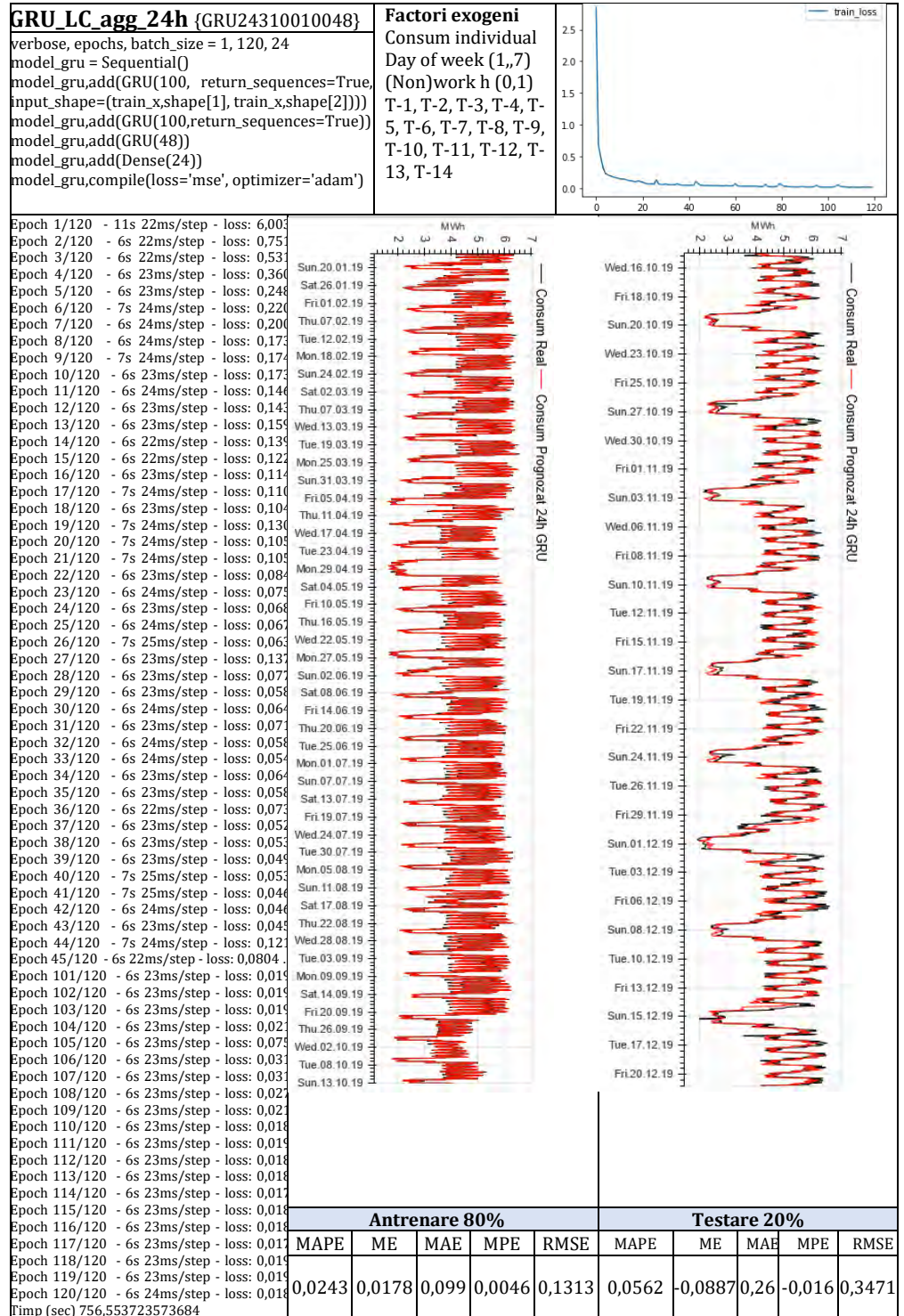
```

Factori exogeni
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14

Epoch 1/50 - 11s 20ms/step - loss: 6,4749
Epoch 2/50 - 6s 21ms/step - loss: 1,0687
Epoch 3/50 - 6s 22ms/step - loss: 0,5727
Epoch 4/50 - 6s 22ms/step - loss: 0,3609
Epoch 5/50 - 6s 22ms/step - loss: 0,2484
Epoch 6/50 - 6s 22ms/step - loss: 0,2166
Epoch 7/50 - 6s 22ms/step - loss: 0,1882
Epoch 8/50 - 6s 22ms/step - loss: 0,1678
Epoch 9/50 - 6s 22ms/step - loss: 0,1726
Epoch 10/50 - 6s 22ms/step - loss: 0,1534
Epoch 11/50 - 6s 22ms/step - loss: 0,1628
Epoch 12/50 - 6s 22ms/step - loss: 0,1419
Epoch 13/50 - 6s 22ms/step - loss: 0,1423
Epoch 14/50 - 6s 22ms/step - loss: 0,1278
Epoch 15/50 - 6s 22ms/step - loss: 0,1302
Epoch 16/50 - 6s 22ms/step - loss: 0,1214
Epoch 17/50 - 6s 22ms/step - loss: 0,1270
Epoch 18/50 - 6s 22ms/step - loss: 0,1039
Epoch 19/50 - 6s 22ms/step - loss: 0,1030
Epoch 20/50 - 6s 22ms/step - loss: 0,1152
Epoch 21/50 - 6s 22ms/step - loss: 0,0994
Epoch 22/50 - 6s 22ms/step - loss: 0,0900
Epoch 23/50 - 6s 22ms/step - loss: 0,0820
Epoch 24/50 - 6s 22ms/step - loss: 0,0712
Epoch 25/50 - 6s 22ms/step - loss: 0,0753
Epoch 26/50 - 6s 22ms/step - loss: 0,1079
Epoch 27/50 - 6s 22ms/step - loss: 0,0918
Epoch 28/50 - 6s 22ms/step - loss: 0,0747
Epoch 29/50 - 6s 22ms/step - loss: 0,0617
Epoch 30/50 - 6s 22ms/step - loss: 0,0620
Epoch 31/50 - 6s 22ms/step - loss: 0,0665
Epoch 32/50 - 6s 22ms/step - loss: 0,0829
Epoch 33/50 - 6s 22ms/step - loss: 0,0578
Epoch 34/50 - 6s 22ms/step - loss: 0,0562
Epoch 35/50 - 6s 22ms/step - loss: 0,0582
Epoch 36/50 - 6s 22ms/step - loss: 0,0605
Epoch 37/50 - 6s 22ms/step - loss: 0,0891
Epoch 38/50 - 6s 22ms/step - loss: 0,0516
Epoch 39/50 - 6s 22ms/step - loss: 0,0559
Epoch 40/50 - 6s 22ms/step - loss: 0,0621
Epoch 41/50 - 6s 22ms/step - loss: 0,0535
Epoch 42/50 - 6s 22ms/step - loss: 0,1513
Epoch 43/50 - 6s 22ms/step - loss: 0,0790
Epoch 44/50 - 6s 22ms/step - loss: 0,0527
Epoch 45/50 - 6s 22ms/step - loss: 0,0505
Epoch 46/50 - 6s 22ms/step - loss: 0,0505
Epoch 47/50 - 6s 22ms/step - loss: 0,0501
Epoch 48/50 - 6s 22ms/step - loss: 0,0528
Epoch 49/50 - 6s 22ms/step - loss: 0,0836
Epoch 50/50 - 6s 22ms/step - loss: 0,0444

Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0434	0,0909	0,172	0,0234	0,2467	0,052	-0,0216	0,2497	-0,0035	0,3364

GRU_LC_agg_24h {GRU24310010048}	Factori exogeni																															
verbose, epochs, batch_size = 1, 100, 24 model_gru = Sequential() model_gru.add(GRU(100, return_sequences=True, input_shape=(train_x_shape[1], train_x_shape[2]))) model_gru.add(GRU(100, return_sequences=True)) model_gru.add(GRU(48)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam')	Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14																															
Epoch 1/100 - 16s 43ms/step - loss: 6,41 Epoch 2/100 - 10s 36ms/step - loss: 1,26 Epoch 3/100 - 10s 38ms/step - loss: 0,58 Epoch 4/100 - 15s 56ms/step - loss: 0,44 Epoch 5/100 - 13s 46ms/step - loss: 0,27 Epoch 6/100 - 13s 46ms/step - loss: 0,21 Epoch 7/100 - 13s 46ms/step - loss: 0,19 Epoch 8/100 - 13s 46ms/step - loss: 0,17 Epoch 9/100 - 13s 47ms/step - loss: 0,17 Epoch 10/100 - 13s 48ms/step - loss: 0,17 Epoch 11/100 - 11s 39ms/step - loss: 0,17 Epoch 12/100 - 14s 50ms/step - loss: 0,15 Epoch 13/100 - 11s 41ms/step - loss: 0,13 Epoch 14/100 - 11s 40ms/step - loss: 0,13 Epoch 15/100 - 11s 40ms/step - loss: 0,12 Epoch 16/100 - 11s 40ms/step - loss: 0,11 Epoch 17/100 - 11s 40ms/step - loss: 0,13 Epoch 18/100 - 11s 40ms/step - loss: 0,11 Epoch 19/100 - 11s 42ms/step - loss: 0,10 Epoch 20/100 - 11s 40ms/step - loss: 0,09 Epoch 21/100 - 11s 40ms/step - loss: 0,10 Epoch 22/100 - 11s 40ms/step - loss: 0,08 Epoch 23/100 - 11s 40ms/step - loss: 0,09 Epoch 24/100 - 11s 40ms/step - loss: 0,08 Epoch 25/100 - 11s 40ms/step - loss: 0,09 Epoch 26/100 - 11s 40ms/step - loss: 0,06 Epoch 27/100 - 11s 40ms/step - loss: 0,11 Epoch 28/100 - 11s 40ms/step - loss: 0,07 Epoch 29/100 - 12s 43ms/step - loss: 0,06 Epoch 30/100 - 11s 41ms/step - loss: 0,05 Epoch 31/100 - 11s 40ms/step - loss: 0,05 Epoch 32/100 - 11s 41ms/step - loss: 0,05 Epoch 33/100 - 11s 41ms/step - loss: 0,05 Epoch 34/100 - 11s 40ms/step - loss: 0,05 Epoch 35/100 - 11s 40ms/step - loss: 0,05 Epoch 36/100 - 11s 40ms/step - loss: 0,11 Epoch 37/100 - 11s 40ms/step - loss: 0,06 Epoch 38/100 - 11s 40ms/step - loss: 0,07 Epoch 39/100 - 11s 40ms/step - loss: 0,04 Epoch 40/100 - 11s 40ms/step - loss: 0,04 Epoch 41/100 - 11s 41ms/step - loss: 0,04 Epoch 42/100 - 11s 40ms/step - loss: 0,04 Epoch 43/100 - 11s 40ms/step - loss: 0,04 Epoch 44/100 - 11s 40ms/step - loss: 0,05 Epoch 45/100 - 11s 40ms/step - loss: 0,04 Epoch 46/100 - 11s 40ms/step - loss: 0,04 Epoch 47/100 - 11s 40ms/step - loss: 0,04 Epoch 48/100 - 11s 40ms/step - loss: 0,10 Epoch 49/100 - 11s 40ms/step - loss: 0,04 Epoch 50/100 - 11s 40ms/step - loss: 0,04 Epoch 51/100 - 11s 40ms/step - loss: 0,05 Epoch 52/100 - 12s 43ms/step - loss: 0,09 Epoch 53/100 - 12s 42ms/step - loss: 0,05 Epoch 54/100 - 12s 43ms/step - loss: 0,04 Epoch 55/100 - 13s 49ms/step - loss: 0,03 Epoch 56/100 - 11s 42ms/step - loss: 0,03 Epoch 57/100 - 11s 40ms/step - loss: 0,03 Epoch 58/100 - 11s 40ms/step - loss: 0,03 Epoch 59/100 - 11s 40ms/step - loss: 0,03 Epoch 60/100 - 11s 40ms/step - loss: 0,0325 Epoch 98/100 - 12s 44ms/step - loss: 0,01 Epoch 99/100 - 10s 39ms/step - loss: 0,01 Epoch 100/100 - 10s 37ms/step - loss: 0,0176																																
		<table border="1"> <thead> <tr> <th colspan="5">Antrenare 80%</th> <th colspan="5">Testare 20%</th> </tr> <tr> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> </tr> </thead> <tbody> <tr> <td>0,03</td> <td>-0,0783</td> <td>0,1238</td> <td>-0,0185</td> <td>0,1575</td> <td>0,057</td> <td>-0,1919</td> <td>0,3144</td> <td>-0,0389</td> <td>0,429</td> </tr> </tbody> </table>	Antrenare 80%					Testare 20%					MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE	0,03	-0,0783	0,1238	-0,0185	0,1575	0,057	-0,1919	0,3144	-0,0389	0,429
Antrenare 80%					Testare 20%																											
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE																							
0,03	-0,0783	0,1238	-0,0185	0,1575	0,057	-0,1919	0,3144	-0,0389	0,429																							



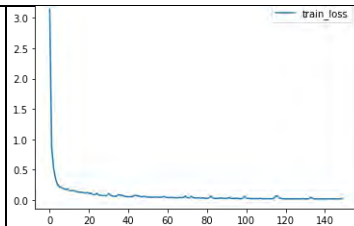
GRU_LC_agg_24h {GRU24310010048}

```

verbose, epochs, batch_size = 1, 150, 24
model_gru = Sequential()
model_gru.add(GRU(100, return_sequences=True,
input_shape=(train_x.shape[1], train_x.shape[2])))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(48))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
    
```

Factori exogeni

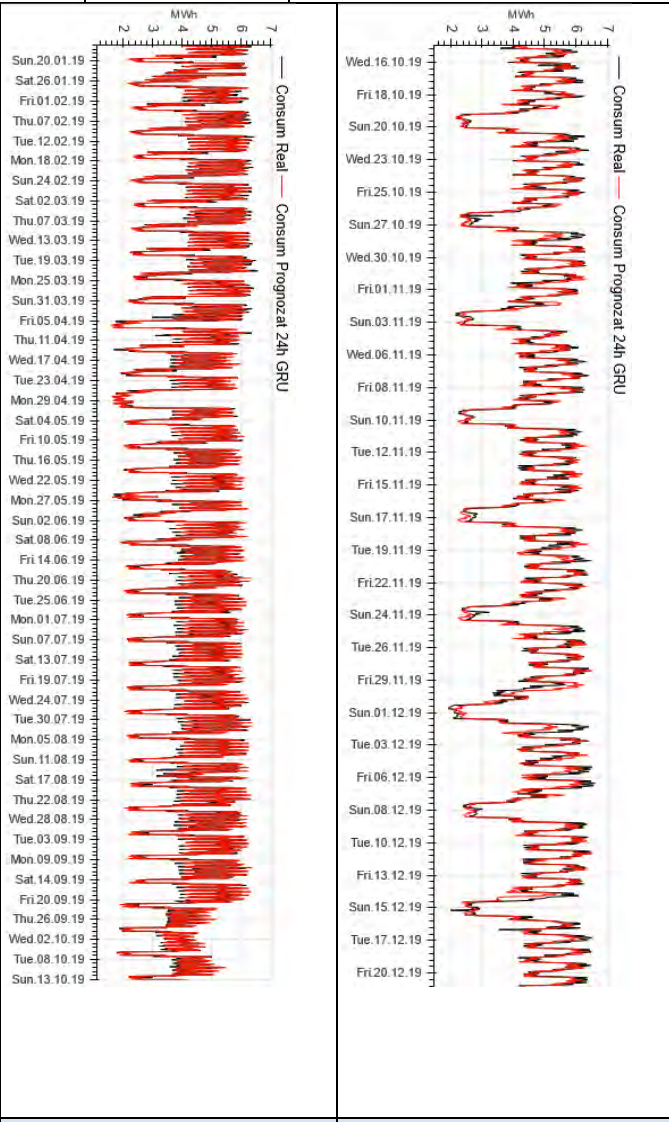
Consum individual
 Day of week (1,7)
 (Non)work h (0,1)
 T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14



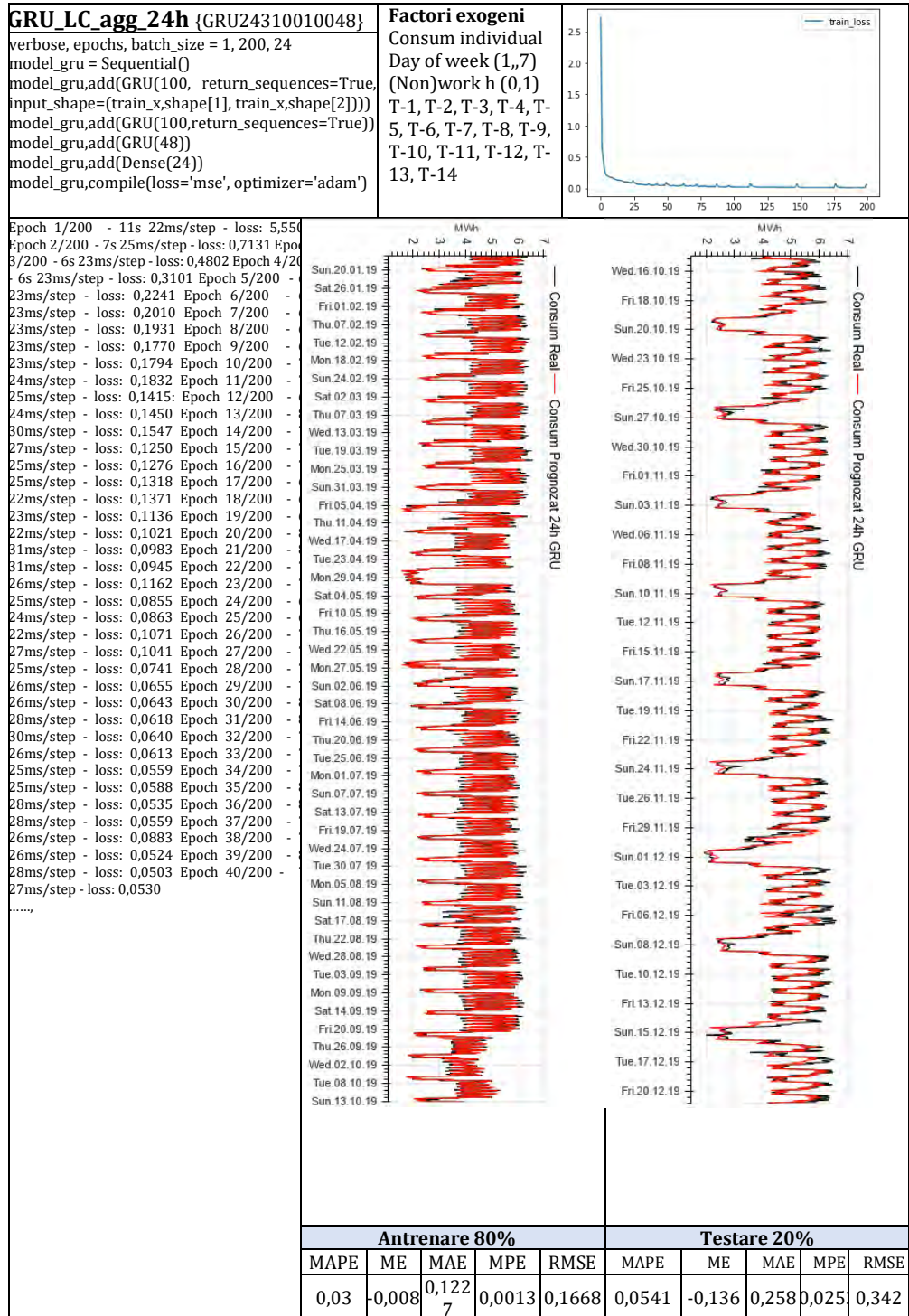
Epoch 1/150 - 11s 23ms/step - loss: 6,377
 Epoch 2/150 - 7s 27ms/step - loss: 1,1242
 Epoch 3/150 - 8s 31ms/step - loss: 0,5420
 Epoch 4/150 - 7s 27ms/step - loss: 0,3859
 Epoch 5/150 - 24ms/step - loss: 0,2639
 Epoch 6/150 - 24ms/step - loss: 0,2290
 Epoch 7/150 - 24ms/step - loss: 0,2097
 Epoch 8/150 - 24ms/step - loss: 0,1911
 Epoch 9/150 - 24ms/step - loss: 0,1774
 Epoch 10/150 - 24ms/step - loss: 0,1970
 Epoch 11/150 - 24ms/step - loss: 0,1673
 Epoch 12/150 - 25ms/step - loss: 0,1453
 Epoch 13/150 - 24ms/step - loss: 0,1485
 Epoch 14/150 - 24ms/step - loss: 0,1467
 Epoch 15/150 - 24ms/step - loss: 0,1407
 Epoch 16/150 - 24ms/step - loss: 0,1341
 Epoch 17/150 - 24ms/step - loss: 0,1165
 Epoch 18/150 - 24ms/step - loss: 0,1376
 Epoch 19/150 - 24ms/step - loss: 0,1114
 Epoch 20/150 - 24ms/step - loss: 0,1244
 Epoch 21/150 - 24ms/step - loss: 0,0998
 Epoch 22/150 - 27ms/step - loss: 0,1259
 Epoch 23/150 - 27ms/step - loss: 0,0888
 Epoch 24/150 - 29ms/step - loss: 0,0874
 Epoch 25/150 - 29ms/step - loss: 0,1068
 Epoch 26/150 - 23ms/step - loss: 0,0800
 Epoch 27/150 - 24ms/step - loss: 0,0789
 Epoch 28/150 - 23ms/step - loss: 0,0746
 Epoch 29/150 - 23ms/step - loss: 0,0712
 Epoch 30/150 - 22ms/step - loss: 0,0698
 Epoch 31/150 - 22ms/step - loss: 0,0928
 Epoch 32/150 - 24ms/step - loss: 0,0865
 Epoch 33/150 - 23ms/step - loss: 0,0668
 Epoch 34/150 - 23ms/step - loss: 0,0652
 Epoch 35/150 - 23ms/step - loss: 0,0682
 Epoch 36/150 - 23ms/step - loss: 0,0639
 Epoch 37/150 - 22ms/step - loss: 0,1041
 Epoch 38/150 - 22ms/step - loss: 0,0630
 Epoch 39/150 - 22ms/step - loss: 0,0623
 Epoch 40/150 - 23ms/step - loss: 0,0553

.....

Epoch 128/150 - 7s 25ms/step - loss: 0,02
 Epoch 129/150 - 7s 24ms/step - loss: 0,018
 Epoch 130/150 - 6s 23ms/step - loss: 0,02
 Epoch 131/150 - 6s 23ms/step - loss: 0,02
 Epoch 132/150 - 6s 23ms/step - loss: 0,01
 Epoch 133/150 - 6s 23ms/step - loss: 0,01
 Epoch 134/150 - 8s 30ms/step - loss: 0,06
 Epoch 135/150 - 7s 25ms/step - loss: 0,01
 Epoch 136/150 - 6s 23ms/step - loss: 0,01
 Epoch 137/150 - 6s 22ms/step - loss: 0,01
 Epoch 138/150 - 9s 32ms/step - loss: 0,01
 Epoch 139/150 - 8s 31ms/step - loss: 0,01
 Epoch 140/150 - 8s 28ms/step - loss: 0,01
 Epoch 141/150 - 9s 32ms/step - loss: 0,01
 Epoch 142/150 - 8s 28ms/step - loss: 0,01
 Epoch 143/150 - 7s 25ms/step - loss: 0,01
 Epoch 144/150 - 7s 25ms/step - loss: 0,01
 Epoch 145/150 - 7s 26ms/step - loss: 0,01
 Epoch 146/150 - 8s 28ms/step - loss: 0,01
 Epoch 147/150 - 8s 29ms/step - loss: 0,01
 Epoch 148/150 - 7s 26ms/step - loss: 0,01
 Epoch 149/150 - 7s 24ms/step - loss: 0,01
 Epoch 150/150 - 7s 27ms/step - loss: 0,02
 Timp (sec) 999,7700130939484



Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0335	0,015	0,131	0,0042	0,1732	0,0588	-0,091	0,267	-0,02	0,3534

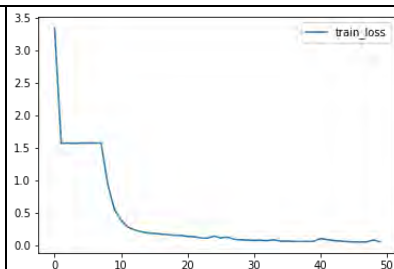


```

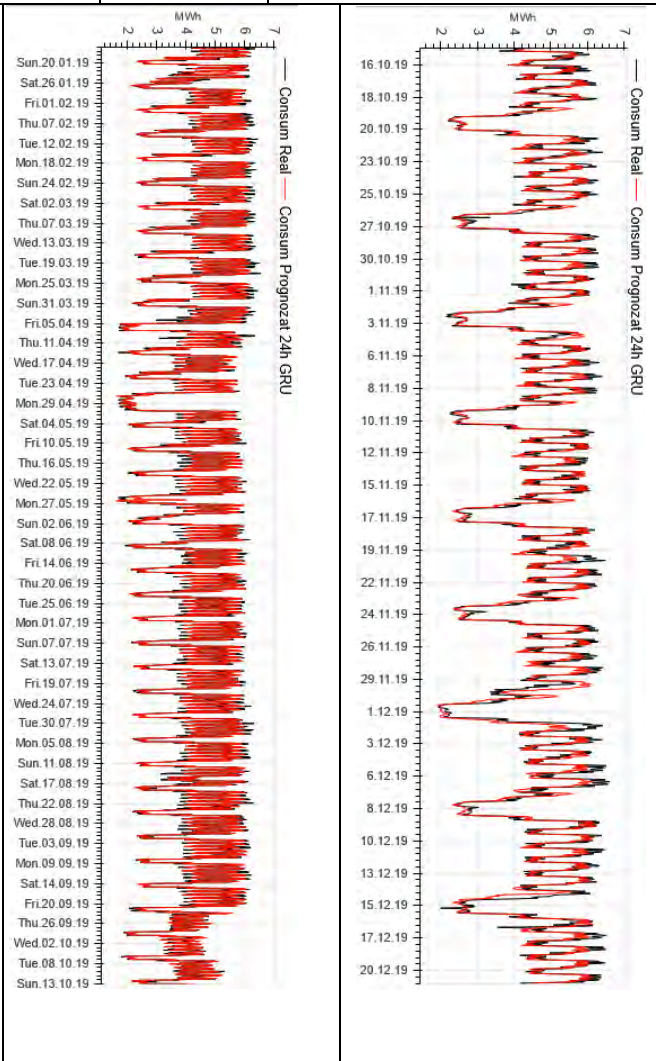
GRU_IC_agg_24h {GRU24524487248}
verbose, epochs, batch_size = 1, 50, 24
model_gru = Sequential()
model_gru.add(GRU(24,return_sequences=True,
input_shape = (train_x.shape[1],train_x.shape[2])))
model_gru.add(GRU(48, return_sequences=True))
model_gru.add(GRU(72, return_sequences=True))
model_gru.add(GRU(72, return_sequences=True))
model_gru.add(GRU(48))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')

```

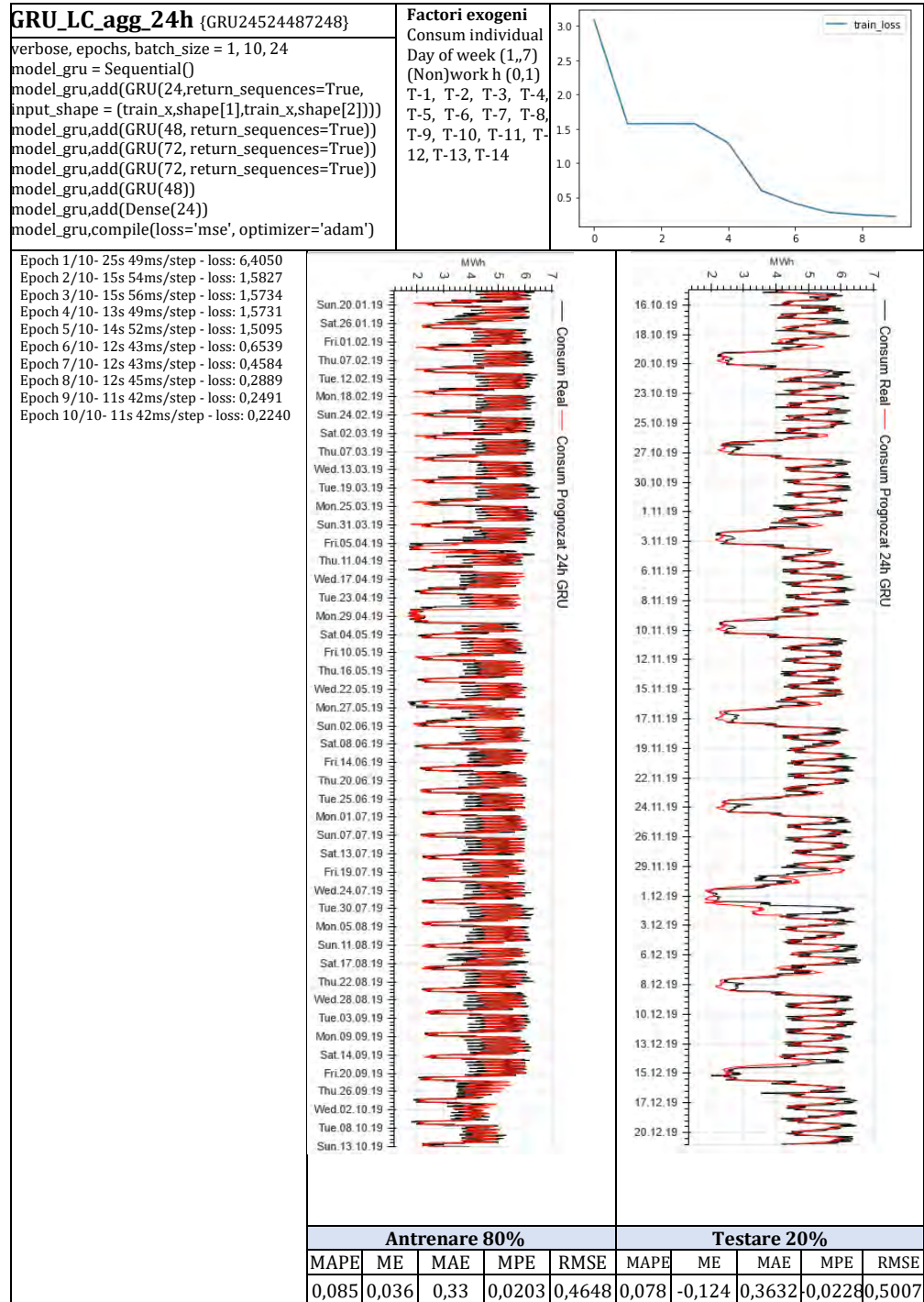
Factori exogeni
Consum individual
Day of week (1,,7)
(Non)work h
(0,1)
T-1, T-2, T-3, T-4,
T-5, T-6, T-7, T-8,
T-9, T-10, T-11, T-
12, T-13, T-14

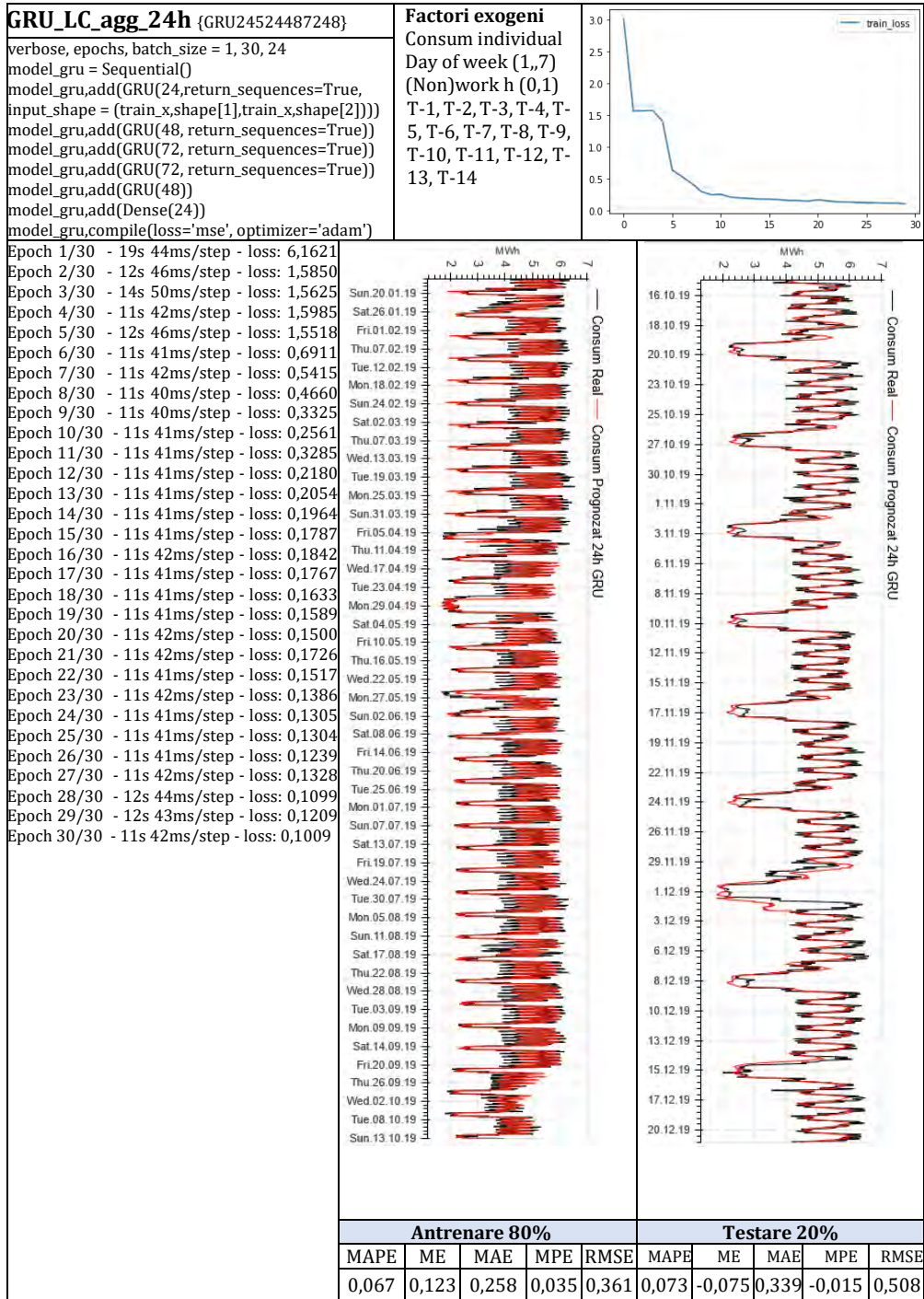


Epoch 1/50 - 19s 40ms/step - loss: 6,9078
Epoch 2/50 - 11s 41ms/step - loss: 1,5956
Epoch 3/50 - 11s 41ms/step - loss: 1,5545
Epoch 4/50 - 11s 41ms/step - loss: 1,5817
Epoch 5/50 - 11s 41ms/step - loss: 1,5721
Epoch 6/50 - 11s 40ms/step - loss: 1,5667
Epoch 7/50 - 11s 40ms/step - loss: 1,5772
Epoch 8/50 - 11s 41ms/step - loss: 1,5671
Epoch 9/50 - 11s 40ms/step - loss: 1,1781
Epoch 10/50 - 11s 40ms/step - loss: 0,6030
Epoch 11/50 - 11s 41ms/step - loss: 0,4107
Epoch 12/50 - 11s 41ms/step - loss: 0,3001
Epoch 13/50 - 11s 41ms/step - loss: 0,2474
Epoch 14/50 - 11s 40ms/step - loss: 0,2183
Epoch 15/50 - 11s 41ms/step - loss: 0,1966
Epoch 16/50 - 11s 40ms/step - loss: 0,1811
Epoch 17/50 - 11s 41ms/step - loss: 0,1764
Epoch 18/50 - 11s 41ms/step - loss: 0,1693
Epoch 19/50 - 11s 41ms/step - loss: 0,1555
Epoch 20/50 - 11s 40ms/step - loss: 0,1658
Epoch 21/50 - 11s 40ms/step - loss: 0,1354
Epoch 22/50 - 11s 41ms/step - loss: 0,1420
Epoch 23/50 - 11s 41ms/step - loss: 0,1138
Epoch 24/50 - 11s 41ms/step - loss: 0,1072
Epoch 25/50 - 11s 40ms/step - loss: 0,1293
Epoch 26/50 - 11s 41ms/step - loss: 0,1184
Epoch 27/50 - 11s 41ms/step - loss: 0,1250
Epoch 28/50 - 13s 48ms/step - loss: 0,0925
Epoch 29/50 - 11s 40ms/step - loss: 0,0823
Epoch 30/50 - 11s 42ms/step - loss: 0,0788
Epoch 31/50 - 11s 42ms/step - loss: 0,0739
Epoch 32/50 - 11s 40ms/step - loss: 0,0729
Epoch 33/50 - 11s 40ms/step - loss: 0,0719
Epoch 34/50 - 11s 40ms/step - loss: 0,0942
Epoch 35/50 - 11s 40ms/step - loss: 0,0631
Epoch 36/50 - 11s 40ms/step - loss: 0,0646
Epoch 37/50 - 11s 40ms/step - loss: 0,0620
Epoch 38/50 - 11s 41ms/step - loss: 0,0563
Epoch 39/50 - 12s 43ms/step - loss: 0,0570
Epoch 40/50 - 11s 42ms/step - loss: 0,0617
Epoch 41/50 - 11s 42ms/step - loss: 0,0758
Epoch 42/50 - 12s 43ms/step - loss: 0,0850
Epoch 43/50 - 11s 41ms/step - loss: 0,0736
Epoch 44/50 - 13s 47ms/step - loss: 0,0621
Epoch 45/50 - 11s 42ms/step - loss: 0,0637
Epoch 46/50 - 11s 42ms/step - loss: 0,0501
Epoch 47/50 - 11s 42ms/step - loss: 0,0508
Epoch 48/50 - 11s 41ms/step - loss: 0,0578
Epoch 49/50 - 11s 42ms/step - loss: 0,0838
Epoch 50/50 - 12s 43ms/step - loss: 0,0573



Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,039	-0,011	0,158	0,0039	0,218	0,059	-0,108	0,279	-0,019	0,366





GRU_LC_agg_24h {GRU24524487248}

```

verbose, epochs, batch_size = 1, 100, 24
model_gru = Sequential()
model_gru.add(GRU(24,return_sequences=True,
input_shape = (train_x_shape[1],train_x_shape[2])))
model_gru.add(GRU(48, return_sequences=True))
model_gru.add(GRU(72, return_sequences=True))
model_gru.add(GRU(72, return_sequences=True))
model_gru.add(GRU(48))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')

```

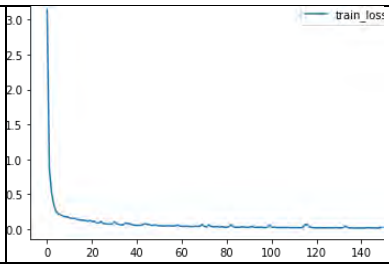
Factori exogeni

Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4,
T-5, T-6, T-7, T-8,
T-9, T-10, T-11, T-12, T-13, T-14

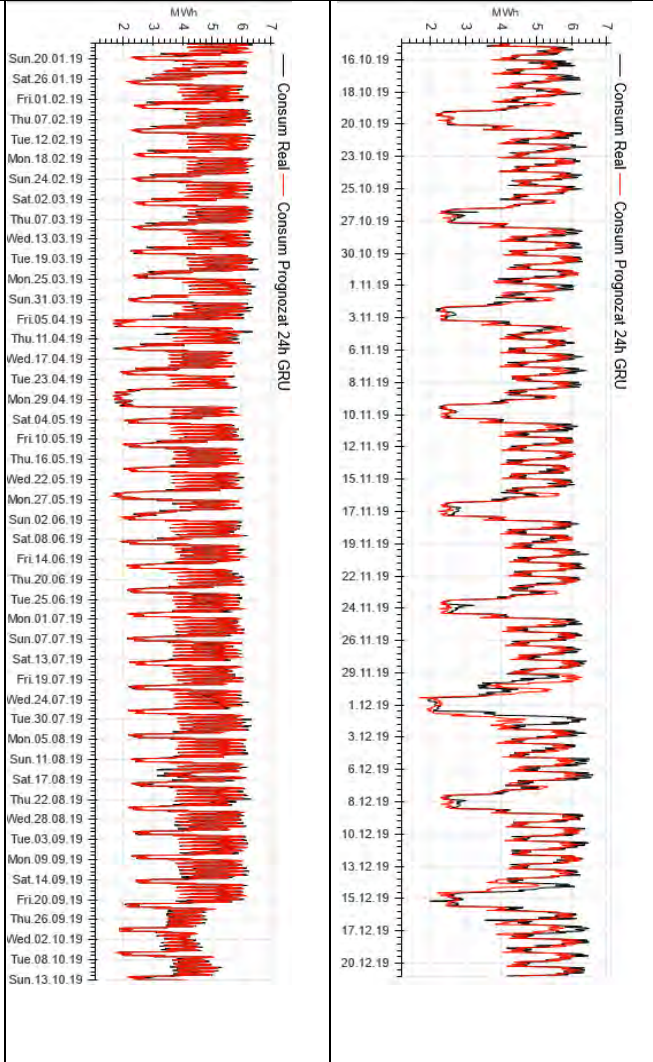
<pre> Epoch 1/100 - 24s 44ms/step - loss: 6,9032 Epoch 2/100 - 12s 43ms/step - loss: 1,5681 Epoch 3/100 - 11s 42ms/step - loss: 1,0453 Epoch 4/100 - 12s 43ms/step - loss: 0,5796 Epoch 5/100 - 11s 42ms/step - loss: 0,4061 Epoch 6/100 - 11s 42ms/step - loss: 0,2833 Epoch 7/100 - 13s 47ms/step - loss: 0,2528 Epoch 8/100 - 11s 42ms/step - loss: 0,2429 Epoch 9/100 - 12s 43ms/step - loss: 0,2092 Epoch 10/100 - 12s 43ms/step - loss: 0,1972 Epoch 11/100 - 11s 41ms/step - loss: 0,1942 Epoch 12/100 - 11s 40ms/step - loss: 0,1865 Epoch 13/100 - 11s 41ms/step - loss: 0,1594 Epoch 14/100 - 11s 41ms/step - loss: 0,1674 Epoch 15/100 - 11s 41ms/step - loss: 0,1504 Epoch 16/100 - 11s 41ms/step - loss: 0,1410 Epoch 17/100 - 11s 40ms/step - loss: 0,1444 Epoch 18/100 - 11s 42ms/step - loss: 0,1375 Epoch 19/100 - 11s 41ms/step - loss: 0,1137 Epoch 20/100 - 11s 41ms/step - loss: 0,1307 Epoch 21/100 - 11s 41ms/step - loss: 0,1748 Epoch 22/100 - 11s 41ms/step - loss: 0,1120 Epoch 23/100 - 11s 42ms/step - loss: 0,0936 Epoch 24/100 - 11s 41ms/step - loss: 0,0977 Epoch 25/100 - 11s 42ms/step - loss: 0,1041 Epoch 26/100 - 11s 40ms/step - loss: 0,1135 Epoch 27/100 - 11s 40ms/step - loss: 0,1150 Epoch 28/100 - 12s 44ms/step - loss: 0,0727 Epoch 29/100 - 12s 46ms/step - loss: 0,0709 Epoch 30/100 - 12s 43ms/step - loss: 0,0715 Epoch 31/100 - 15s 54ms/step - loss: 0,0636 Epoch 32/100 - 15s 55ms/step - loss: 0,0714 Epoch 33/100 - 14s 50ms/step - loss: 0,0628 Epoch 34/100 - 13s 50ms/step - loss: 0,0657 Epoch 35/100 - 11s 42ms/step - loss: 0,1168 Epoch 36/100 - 12s 44ms/step - loss: 0,0864 Epoch 37/100 - 11s 41ms/step - loss: 0,0676 Epoch 38/100 - 12s 46ms/step - loss: 0,0872 Epoch 39/100 - 11s 40ms/step - loss: 0,1185 Epoch 40/100 - 11s 40ms/step - loss: 0,0686 Epoch 41/100 - 11s 40ms/step - loss: 0,0542 Epoch 42/100 - 11s 41ms/step - loss: 0,0527 Epoch 43/100 - 11s 41ms/step - loss: 0,0500 Epoch 44/100 - 11s 40ms/step - loss: 0,0552 Epoch 45/100 - 11s 40ms/step - loss: 0,0517 Epoch 46/100 - 12s 44ms/step - loss: 0,0477 Epoch 47/100 - 11s 40ms/step - loss: 0,0462 Epoch 48/100 - 11s 41ms/step - loss: 0,0520 Epoch 49/100 - 11s 41ms/step - loss: 0,1081 Epoch 50/100 - 13s 48ms/step - loss: 0,0656 Epoch 51/100 - 12s 46ms/step - loss: 0,0455 Epoch 52/100 - 11s 40ms/step - loss: 0,0441 Epoch 53/100 - 11s 40ms/step - loss: 0,0438 Epoch 54/100 - 11s 41ms/step - loss: 0,0424 Epoch 55/100 - 11s 39ms/step - loss: 0,0424 Epoch 56/100 - 11s 39ms/step - loss: 0,0451 Epoch 57/100 - 11s 39ms/step - loss: 0,0400 Epoch 58/100 - 11s 39ms/step - loss: 0,0413 Epoch 59/100 - 11s 39ms/step - loss: 0,0412 Epoch 60/100 - 11s 39ms/step - loss: 0,0204 Epoch 98/100 - 11s 39ms/step - loss: 0,0209 Epoch 99/100 - 11s 39ms/step - loss: 0,0213 Epoch 100/100 - 11s 39ms/step - loss: 0,0195 </pre>																															
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="5">Antrenare 80%</th> <th colspan="5">Testare 20%</th> </tr> <tr> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> </tr> </thead> <tbody> <tr> <td>0,026</td> <td>0,005</td> <td>0,106</td> <td>0,003</td> <td>0,141</td> <td>0,063</td> <td>-0,103</td> <td>0,291</td> <td>-0,019</td> <td>0,408</td> </tr> </tbody> </table>	Antrenare 80%					Testare 20%					MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE	0,026	0,005	0,106	0,003	0,141	0,063	-0,103	0,291	-0,019	0,408
Antrenare 80%					Testare 20%																										
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE																						
0,026	0,005	0,106	0,003	0,141	0,063	-0,103	0,291	-0,019	0,408																						

```
GRU_IC_agg_24h {GRU24524487248}
verbose, epochs, batch_size = 1, 150, 24
model_gru = Sequential()
model_gru.add(GRU(24,return_sequences=True,
input_shape = (train_x.shape[1],train_x.shape[2])))
model_gru.add(GRU(48, return_sequences=True))
model_gru.add(GRU(72, return_sequences=True))
model_gru.add(GRU(72, return_sequences=True))
model_gru.add(GRU(48))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
```

Factori exogeni
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4,
T-5, T-6, T-7, T-8,
T-9, T-10, T-11, T-
12, T-13, T-14



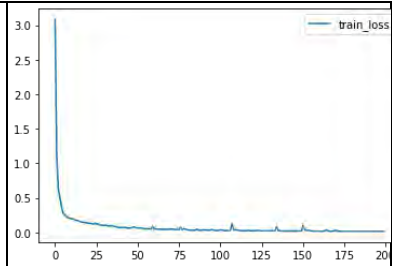
Epoch 1/100 - 24s 44ms/step - loss: 6,9032
Epoch 2/100 - 12s 43ms/step - loss: 1,5681
Epoch 3/100 - 11s 42ms/step - loss: 1,0453
Epoch 4/100 - 12s 43ms/step - loss: 0,5796
Epoch 5/100 - 11s 42ms/step - loss: 0,4061
Epoch 6/100 - 11s 42ms/step - loss: 0,2833
Epoch 7/100 - 13s 47ms/step - loss: 0,2528
Epoch 8/100 - 11s 42ms/step - loss: 0,2429
Epoch 9/100 - 12s 43ms/step - loss: 0,2092
Epoch 10/100 - 12s 43ms/step - loss: 0,1972
Epoch 11/100 - 11s 41ms/step - loss: 0,1942
Epoch 12/100 - 11s 40ms/step - loss: 0,1865
Epoch 13/100 - 11s 41ms/step - loss: 0,1594
Epoch 14/100 - 11s 41ms/step - loss: 0,1674
Epoch 15/100 - 11s 41ms/step - loss: 0,1504
Epoch 16/100 - 11s 41ms/step - loss: 0,1410
Epoch 17/100 - 11s 40ms/step - loss: 0,1444
Epoch 18/100 - 11s 42ms/step - loss: 0,1375
Epoch 19/100 - 11s 41ms/step - loss: 0,1137
Epoch 20/100 - 11s 41ms/step - loss: 0,1307
Epoch 21/100 - 11s 41ms/step - loss: 0,1748
Epoch 22/100 - 11s 41ms/step - loss: 0,1120
Epoch 23/100 - 11s 42ms/step - loss: 0,0936
Epoch 24/100 - 11s 41ms/step - loss: 0,0977
Epoch 25/100 - 11s 42ms/step - loss: 0,1041
Epoch 26/100 - 11s 40ms/step - loss: 0,1135
Epoch 27/100 - 11s 40ms/step - loss: 0,1150
Epoch 28/100 - 12s 44ms/step - loss: 0,0727
Epoch 29/100 - 12s 46ms/step - loss: 0,0709
Epoch 30/100 - 12s 43ms/step - loss: 0,0715
Epoch 31/100 - 15s 54ms/step - loss: 0,0636
Epoch 32/100 - 15s 55ms/step - loss: 0,0714
Epoch 33/100 - 14s 50ms/step - loss: 0,0628
Epoch 34/100 - 13s 50ms/step - loss: 0,0657
Epoch 35/100 - 11s 42ms/step - loss: 0,1168
Epoch 36/100 - 12s 44ms/step - loss: 0,0864
Epoch 37/100 - 11s 41ms/step - loss: 0,0676
Epoch 38/100 - 12s 46ms/step - loss: 0,0872
Epoch 39/100 - 11s 40ms/step - loss: 0,1185
Epoch 40/100 - 11s 40ms/step - loss: 0,0686
.....
Epoch 130/150 - 6s 23ms/step - loss: 0,0220
Epoch 131/150 - 6s 23ms/step - loss: 0,0202
Epoch 132/150 - 6s 23ms/step - loss: 0,0176
Epoch 133/150 - 6s 23ms/step - loss: 0,0170
Epoch 134/150 - 8s 30ms/step - loss: 0,0667
Epoch 135/150 - 7s 25ms/step - loss: 0,0191
Epoch 136/150 - 6s 23ms/step - loss: 0,0165
Epoch 137/150 - 6s 22ms/step - loss: 0,0163
Epoch 138/150 - 9s 32ms/step - loss: 0,0160
Epoch 139/150 - 8s 31ms/step - loss: 0,0163
Epoch 140/150 - 8s 28ms/step - loss: 0,0171
Epoch 141/150 - 9s 32ms/step - loss: 0,0169
Epoch 142/150 - 8s 28ms/step - loss: 0,0165
Epoch 143/150 - 7s 25ms/step - loss: 0,0163
Epoch 144/150 - 7s 25ms/step - loss: 0,0178
Epoch 145/150 - 7s 26ms/step - loss: 0,0176
Epoch 146/150 - 8s 28ms/step - loss: 0,0184
Epoch 147/150 - 8s 29ms/step - loss: 0,0167
Epoch 148/150 - 7s 26ms/step - loss: 0,0165
Epoch 149/150 - 7s 24ms/step - loss: 0,0157
Epoch 150/150 - 7s 27ms/step - loss: 0,0215
Timp (sec) 999,7700130939484



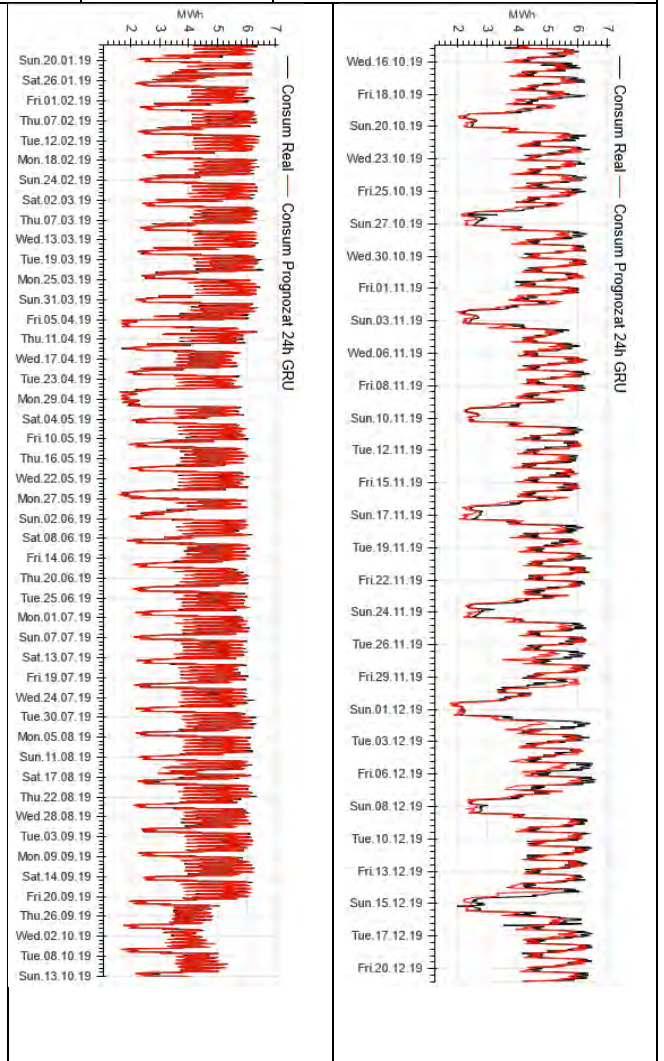
Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0465	-0,006	0,173	0,008	0,272	0,0606	-0,103	0,276	-0,014	0,36

```
GRU_LC_agg_24h {GRU24524487248}
verbose, epochs, batch_size = 1, 200, 24
model_gru = Sequential()
model_gru.add(GRU(24,return_sequences=True,
input_shape = (train_x_shape[1],train_x_shape[2])))
model_gru.add(GRU(48, return_sequences=True))
model_gru.add(GRU(72, return_sequences=True))
model_gru.add(GRU(72, return_sequences=True))
model_gru.add(GRU(48))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
```

Factori exogeni
Consum individual
Day of week (1,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14



Epoch 1/200 - 13s 22ms/step - loss: 6,3281
Epoch 2/200 - 7s 25ms/step - loss: 1,3340
Epoch 3/200 - 6s 23ms/step - loss: 0,6788
Epoch 4/200 - 6s 21ms/step - loss: 0,5141
Epoch 5/200 - 6s 21ms/step - loss: 0,3964
Epoch 6/200 - 6s 21ms/step - loss: 0,2930
Epoch 7/200 - 6s 21ms/step - loss: 0,3116
Epoch 8/200 - 6s 22ms/step - loss: 0,2227
Epoch 9/200 - 6s 22ms/step - loss: 0,2127
Epoch 10/200 - 6s 22ms/step - loss: 0,1970
Epoch 11/200 - 6s 22ms/step - loss: 0,2011
Epoch 12/200 - 6s 23ms/step - loss: 0,2168
Epoch 13/200 - 6s 22ms/step - loss: 0,1901
Epoch 14/200 - 6s 22ms/step - loss: 0,1706
Epoch 15/200 - 6s 22ms/step - loss: 0,1686
Epoch 16/200 - 6s 23ms/step - loss: 0,1580
Epoch 17/200 - 6s 22ms/step - loss: 0,1453
Epoch 18/200 - 6s 23ms/step - loss: 0,1442
Epoch 19/200 - 6s 21ms/step - loss: 0,1482
Epoch 20/200 - 6s 21ms/step - loss: 0,1440
Epoch 21/200 - 6s 22ms/step - loss: 0,1400
Epoch 22/200 - 6s 22ms/step - loss: 0,1323
Epoch 23/200 - 6s 23ms/step - loss: 0,1275
Epoch 24/200 - 6s 24ms/step - loss: 0,1213
Epoch 25/200 - 6s 22ms/step - loss: 0,1429
Epoch 26/200 - 6s 22ms/step - loss: 0,1122
Epoch 27/200 - 6s 22ms/step - loss: 0,1492
Epoch 28/200 - 6s 22ms/step - loss: 0,1006
Epoch 29/200 - 6s 22ms/step - loss: 0,1030
Epoch 30/200 - 6s 22ms/step - loss: 0,0996
Epoch 31/200 - 6s 22ms/step - loss: 0,1033
Epoch 32/200 - 6s 22ms/step - loss: 0,0997
Epoch 33/200 - 6s 22ms/step - loss: 0,1025
Epoch 34/200 - 6s 22ms/step - loss: 0,0864
Epoch 35/200 - 6s 21ms/step - loss: 0,0980
Epoch 36/200 - 6s 23ms/step - loss: 0,0972
Epoch 37/200 - 6s 22ms/step - loss: 0,0912
Epoch 38/200 - 6s 22ms/step - loss: 0,0766
Epoch 39/200 - 6s 22ms/step - loss: 0,0757
Epoch 40/200 - 6s 22ms/step - loss: 0,0730
.....
Epoch 180/200 - 6s 24ms/step - loss: 0,0157
Epoch 181/200 - 7s 25ms/step - loss: 0,0172
Epoch 182/200 - 7s 24ms/step - loss: 0,0199
Epoch 183/200 - 7s 24ms/step - loss: 0,0165
Epoch 184/200 - 7s 25ms/step - loss: 0,0170
Epoch 185/200 - 7s 24ms/step - loss: 0,0149
Epoch 186/200 - 7s 25ms/step - loss: 0,0164
Epoch 187/200 - 7s 25ms/step - loss: 0,0145
Epoch 188/200 - 7s 25ms/step - loss: 0,0156
Epoch 189/200 - 7s 24ms/step - loss: 0,0152
Epoch 190/200 - 7s 25ms/step - loss: 0,0155
Epoch 191/200 - 7s 25ms/step - loss: 0,0164
Epoch 192/200 - 6s 23ms/step - loss: 0,0147
Epoch 193/200 - 7s 27ms/step - loss: 0,0155
Epoch 194/200 - 7s 25ms/step - loss: 0,0142
Epoch 195/200 - 7s 28ms/step - loss: 0,0145
Epoch 196/200 - 6s 24ms/step - loss: 0,0158
Epoch 197/200 - 7s 25ms/step - loss: 0,0143
Epoch 198/200 - 6s 24ms/step - loss: 0,0250
Epoch 199/200 - 7s 24ms/step - loss: 0,0135
Epoch 200/200 - 7s 25ms/step - loss: 0,0135
Timp (sec) 1277,180551290512



Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0217	-0,019	0,090	-0,004	0,117	0,065	-0,177	0,302	-0,038	0,405

GRU_IC_agg_24h {GRU 24 3 240 240 168 }		Factori exogeni		Consum individual		Day of week (1,,7)		(Non)work h (0,1)		T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14				
<pre> verbose, epochs, batch_size = 1, 100, 24 model_gru = Sequential() model_gru.add(GRU(240, return_sequences=True,input_shape=(train_x.shape[1], train_x.shape[2]))) model_gru.add(GRU(240, return_sequences=True)) model_gru.add(GRU(168, return_sequences=False)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam') </pre>														
<pre> Epoch 1/100 - 19s 51ms/step - loss: 2,7294 Epoch 2/100 - 14s 52ms/step - loss: 0,5514 Epoch 3/100 - 14s 50ms/step - loss: 0,2606 Epoch 4/100 - 14s 50ms/step - loss: 0,2063 Epoch 5/100 - 15s 54ms/step - loss: 0,1857 Epoch 6/100 - 15s 56ms/step - loss: 0,1723 Epoch 7/100 - 14s 52ms/step - loss: 0,1510 Epoch 8/100 - 15s 54ms/step - loss: 0,1504 Epoch 9/100 - 13s 50ms/step - loss: 0,1179 Epoch 10/100 - 14s 53ms/step - loss: 0,1099 Epoch 11/100 - 15s 55ms/step - loss: 0,0932 Epoch 12/100 - 14s 51ms/step - loss: 0,0916 Epoch 13/100 - 13s 49ms/step - loss: 0,0954 Epoch 14/100 - 14s 50ms/step - loss: 0,1103 Epoch 15/100 - 14s 51ms/step - loss: 0,0747 Epoch 16/100 - 17s 62ms/step - loss: 0,0846 Epoch 17/100 - 16s 58ms/step - loss: 0,0645 Epoch 18/100 - 14s 53ms/step - loss: 0,0672 Epoch 19/100 - 14s 50ms/step - loss: 0,0967 Epoch 20/100 - 14s 50ms/step - loss: 0,0875 Epoch 21/100 - 13s 50ms/step - loss: 0,0510 Epoch 22/100 - 14s 51ms/step - loss: 0,0509 Epoch 23/100 - 14s 52ms/step - loss: 0,0656 Epoch 24/100 - 14s 52ms/step - loss: 0,0631 Epoch 25/100 - 14s 51ms/step - loss: 0,0473 Epoch 26/100 - 14s 53ms/step - loss: 0,0488 Epoch 27/100 - 14s 50ms/step - loss: 0,0388 Epoch 28/100 - 13s 49ms/step - loss: 0,0447 Epoch 29/100 - 13s 50ms/step - loss: 0,0402 Epoch 30/100 - 14s 52ms/step - loss: 0,0409 Epoch 31/100 - 14s 50ms/step - loss: 0,0371 Epoch 32/100 - 13s 49ms/step - loss: 0,0373 Epoch 33/100 - 13s 50ms/step - loss: 0,0391 Epoch 34/100 - 15s 54ms/step - loss: 0,0615 Epoch 35/100 - 15s 56ms/step - loss: 0,0586 Epoch 36/100 - 18s 65ms/step - loss: 0,0515 Epoch 37/100 - 15s 54ms/step - loss: 0,0565 Epoch 38/100 - 14s 51ms/step - loss: 0,0304 Epoch 39/100 - 15s 55ms/step - loss: 0,0359 Epoch 40/100 - 15s 54ms/step - loss: 0,0299 Epoch 41/100 - 15s 55ms/step - loss: 0,0492 Epoch 42/100 - 15s 54ms/step - loss: 0,0538 Epoch 43/100 - 14s 50ms/step - loss: 0,0470 Epoch 44/100 - 13s 49ms/step - loss: 0,0476 Epoch 45/100 - 13s 50ms/step - loss: 0,0316 Epoch 46/100 - 13s 49ms/step - loss: 0,0301 Epoch 47/100 - 13s 50ms/step - loss: 0,0269 Epoch 48/100 - 13s 50ms/step - loss: 0,0253 Epoch 49/100 - 14s 52ms/step - loss: 0,0525 Epoch 50/100 - 14s 52ms/step - loss: 0,0324 Epoch 51/100 - 14s 53ms/step - loss: 0,0217 Epoch 52/100 - 14s 51ms/step - loss: 0,0221 Epoch 53/100 - 14s 50ms/step - loss: 0,0198 Epoch 54/100 - 14s 51ms/step - loss: 0,0183 Epoch 55/100 - 14s 52ms/step - loss: 0,0607 Epoch 56/100 - 14s 52ms/step - loss: 0,0241 Epoch 57/100 - 17s 61ms/step - loss: 0,0184 Epoch 58/100 - 16s 60ms/step - loss: 0,0204 Epoch 59/100 - 14s 51ms/step - loss: 0,0175 Epoch 60/100 - 15s 56ms/step - loss: 0,0157 ... Epoch 98/100 - 14s 53ms/step - loss: 0,0108 Epoch 99/100 - 15s 54ms/step - loss: 0,0101 Epoch 100/100 - 14s 53ms/step - loss: 0,0094 </pre>														
		Antrenare 80%					Testare 20%							
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0184	-0,008	0,075	-0,0023	0,0965	0,0613	-0,129	0,287	-0,023	0,392					

GRU_IC_agg_24h {GRU[24|3|240|240|168]}

```

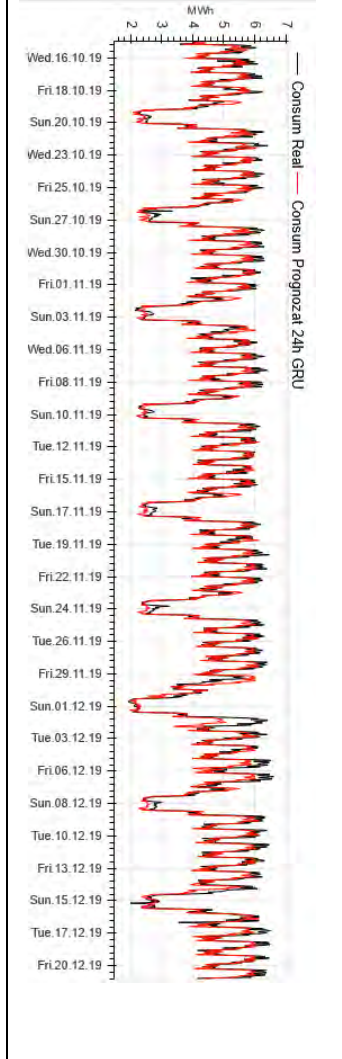
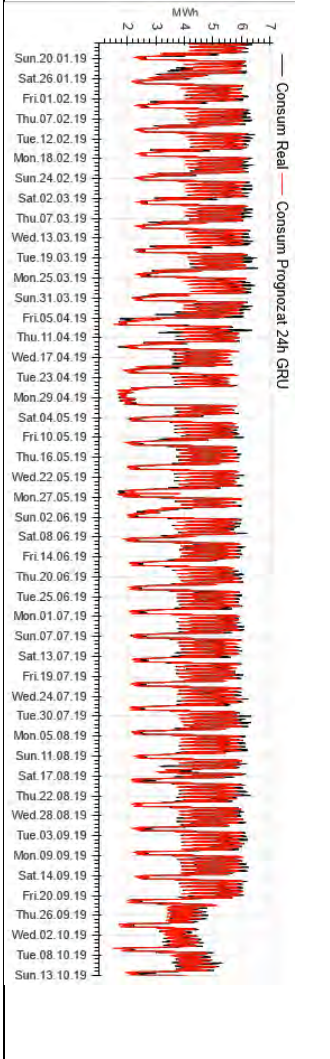
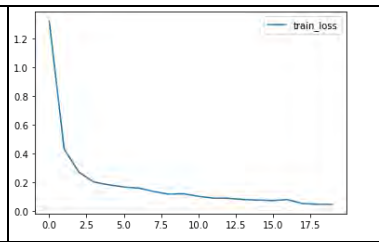
verbose, epochs, batch_size = 1, 20, 24
model_gru = Sequential()
model_gru.add(GRU(240,
return_sequences=True,input_shape=(train_x.shape[1], train_x.shape[2])))
model_gru.add(GRU(240, return_sequences=True))
model_gru.add(GRU(168, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')

```

Epoch 1/20 - 23s 65ms/step - loss: 2,6459
Epoch 2/20 - 19s 69ms/step - loss: 0,5267
Epoch 3/20 - 15s 56ms/step - loss: 0,3529
Epoch 4/20 - 15s 56ms/step - loss: 0,2099
Epoch 5/20 - 15s 56ms/step - loss: 0,1899
Epoch 6/20 - 15s 55ms/step - loss: 0,1599
Epoch 7/20 - 15s 55ms/step - loss: 0,1528
Epoch 8/20 - 15s 56ms/step - loss: 0,1447
Epoch 9/20 - 15s 55ms/step - loss: 0,1202
Epoch 10/20 - 15s 56ms/step - loss: 0,1384
Epoch 11/20 - 15s 56ms/step - loss: 0,1100
Epoch 12/20 - 15s 56ms/step - loss: 0,0916
Epoch 13/20 - 15s 56ms/step - loss: 0,1152
Epoch 14/20 - 15s 56ms/step - loss: 0,0823
Epoch 15/20 - 15s 55ms/step - loss: 0,0819
Epoch 16/20 - 15s 56ms/step - loss: 0,0629
Epoch 17/20 - 15s 57ms/step - loss: 0,0797
Epoch 18/20 - 15s 56ms/step - loss: 0,0546
Epoch 19/20 - 15s 57ms/step - loss: 0,0456
Epoch 20/20 - 15s 56ms/step - loss: 0,0476
Timp (sec) 314,84982323646545

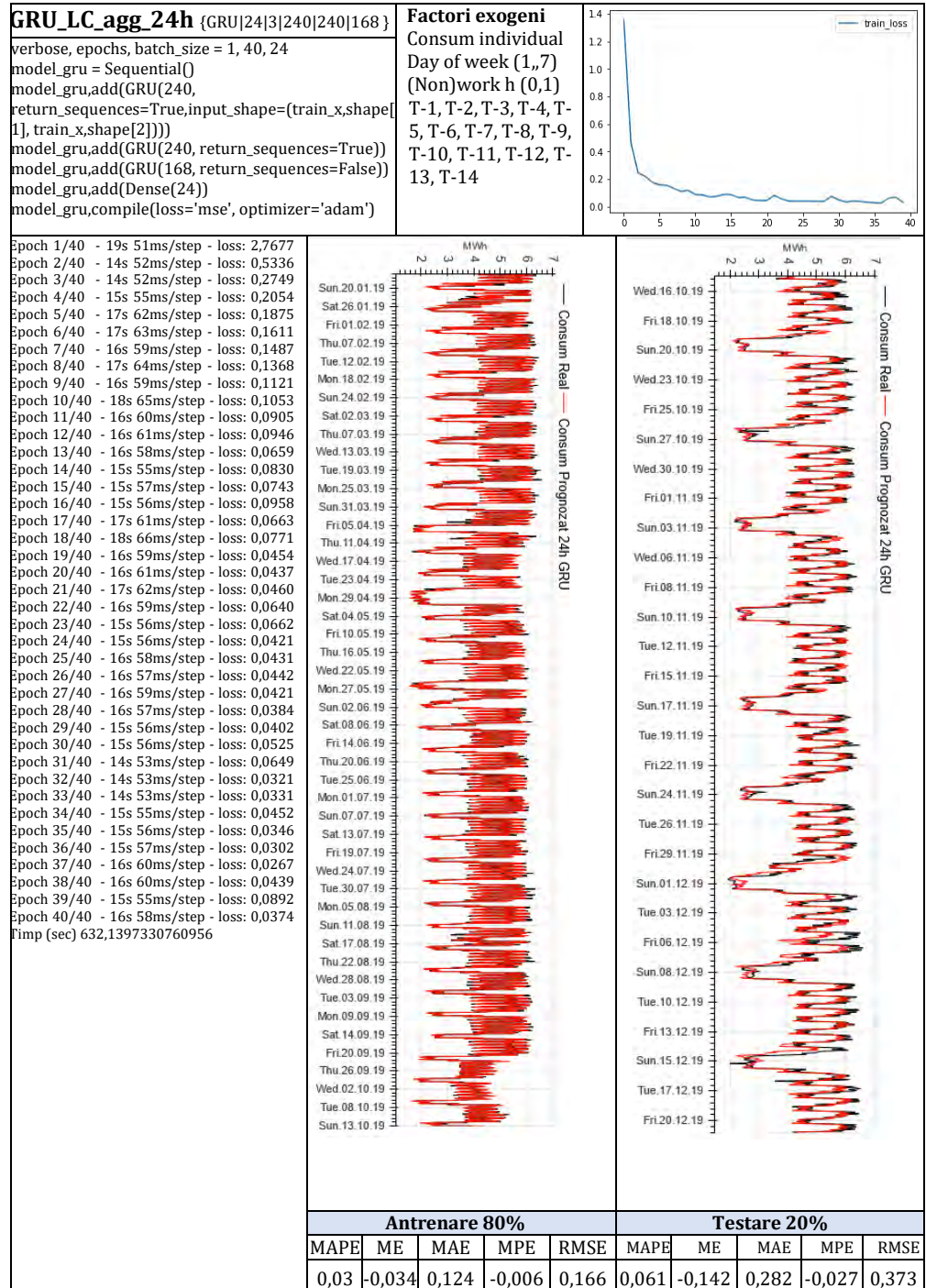
Factori exogeni

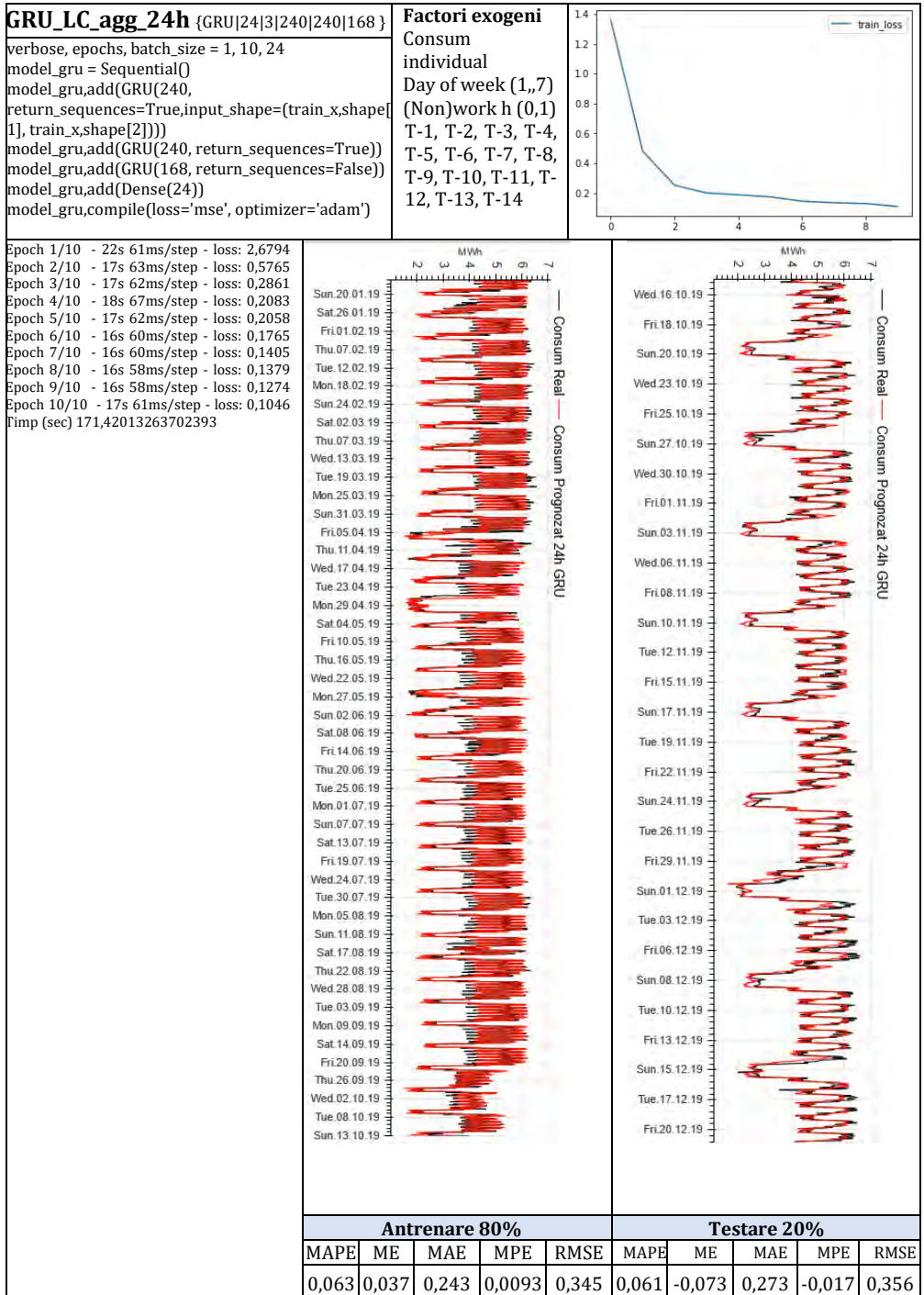
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14

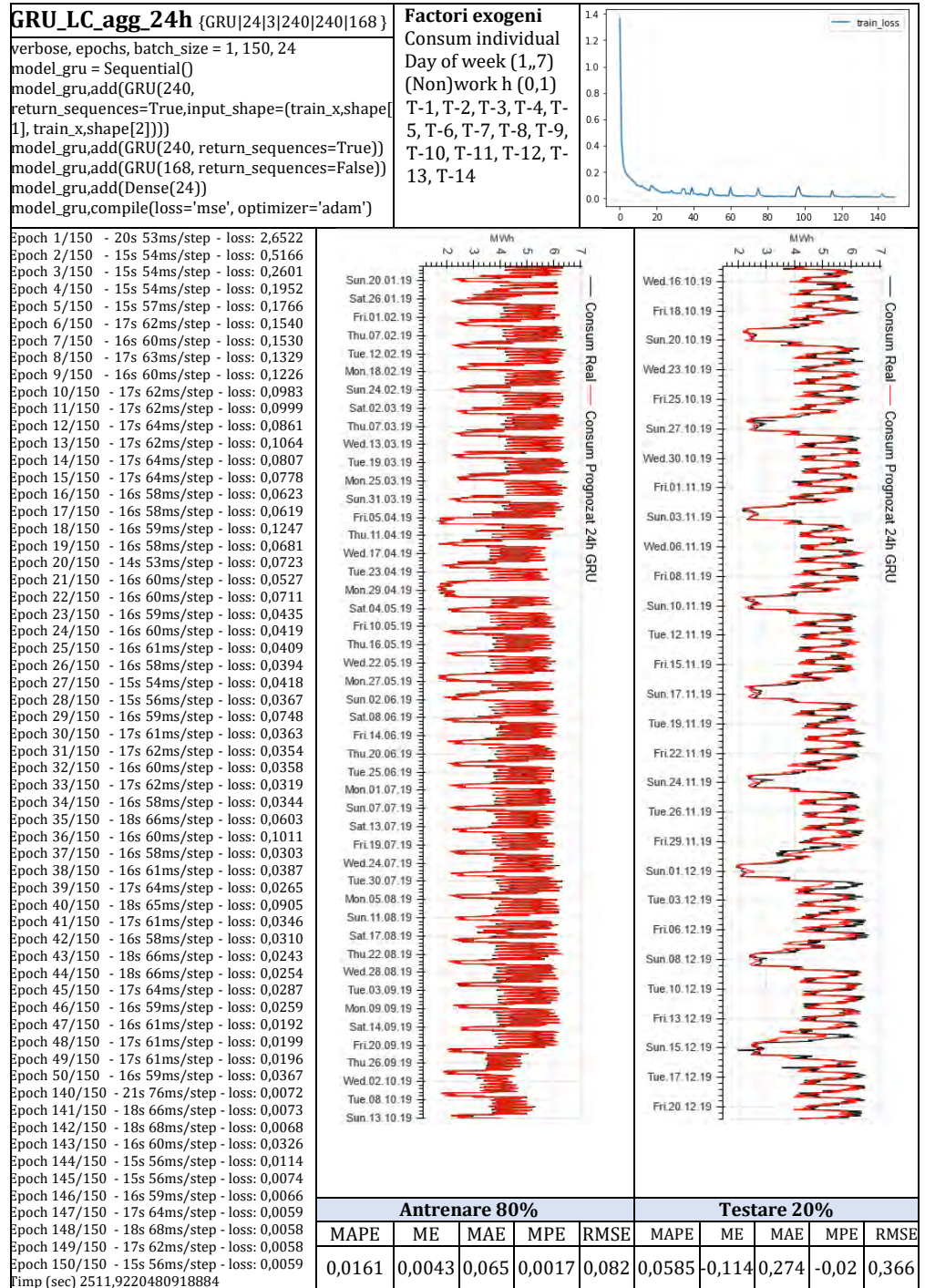


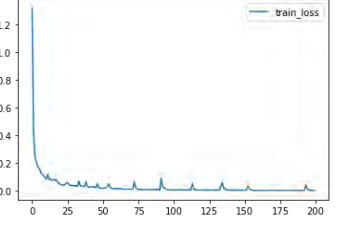
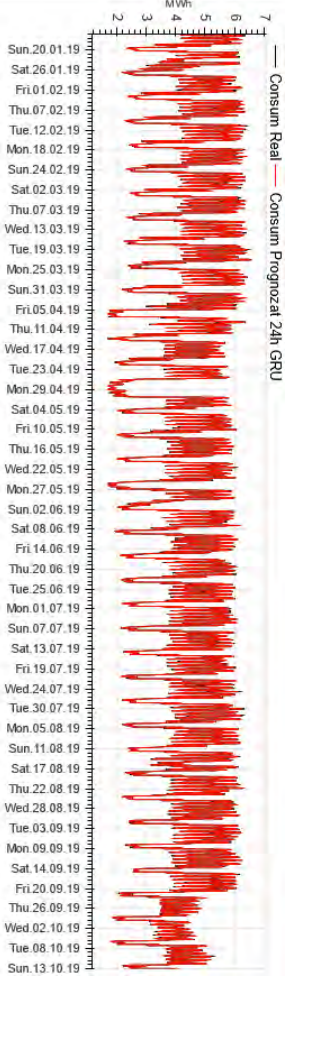
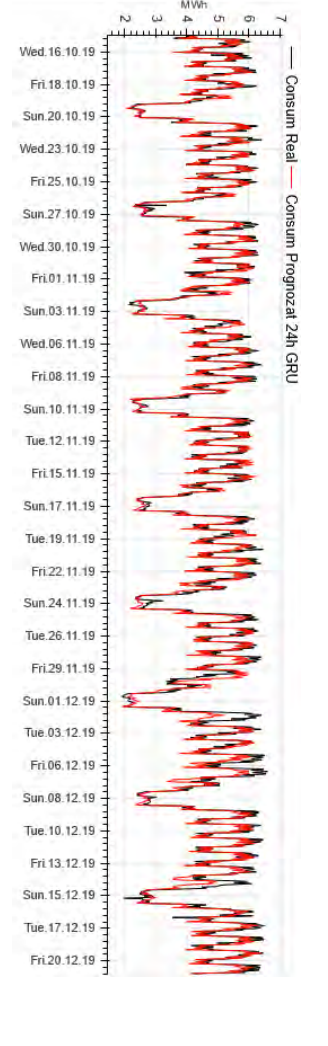
Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,041	-0,036	0,164	-0,006	0,229	0,057	-0,116	0,266	-0,023	0,353

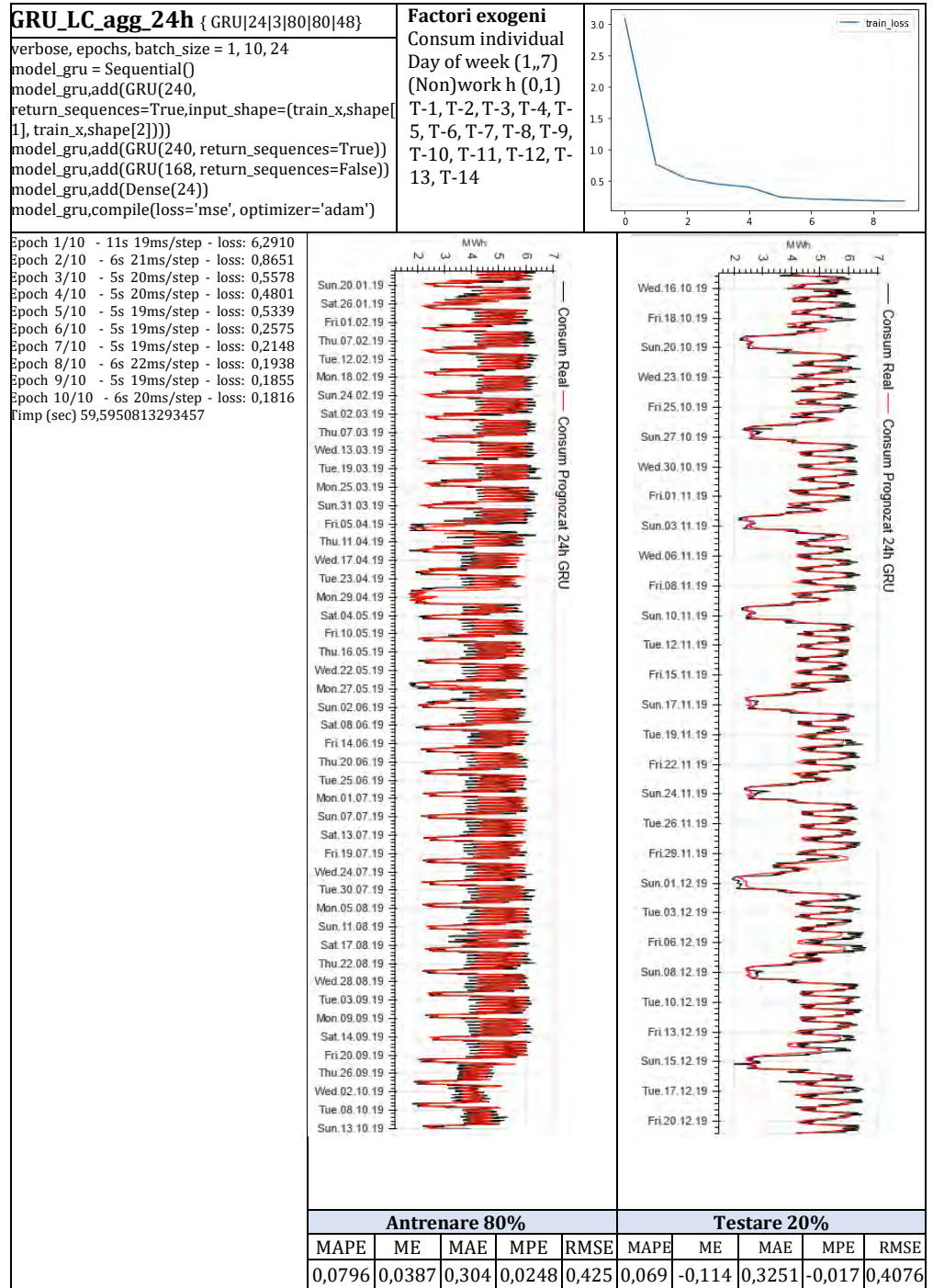
GRU_LC_agg_24h {GRU[24 3 240 240 168]} verbose, epochs, batch_size = 1, 30, 24 model_gru = Sequential() model_gru.add(GRU(240, return_sequences=True,input_shape=(train_x.shape[1], train_x.shape[2]))) model_gru.add(GRU(240, return_sequences=True)) model_gru.add(GRU(168, return_sequences=False)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam')	Factori exogeni Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14																																
Epoch 1/40 - 19s 51ms/step - loss: 2,7677 Epoch 2/40 - 14s 52ms/step - loss: 0,5336 Epoch 3/40 - 14s 52ms/step - loss: 0,2749 Epoch 4/40 - 15s 55ms/step - loss: 0,2054 Epoch 5/40 - 17s 62ms/step - loss: 0,1875 Epoch 6/40 - 17s 63ms/step - loss: 0,1611 Epoch 7/40 - 16s 59ms/step - loss: 0,1487 Epoch 8/40 - 17s 64ms/step - loss: 0,1368 Epoch 9/40 - 16s 59ms/step - loss: 0,1121 Epoch 10/40 - 18s 65ms/step - loss: 0,1053 Epoch 11/40 - 16s 60ms/step - loss: 0,0905 Epoch 12/40 - 16s 61ms/step - loss: 0,0946 Epoch 13/40 - 16s 58ms/step - loss: 0,0659 Epoch 14/40 - 15s 55ms/step - loss: 0,0830 Epoch 15/40 - 15s 57ms/step - loss: 0,0743 Epoch 16/40 - 15s 56ms/step - loss: 0,0958 Epoch 17/40 - 17s 61ms/step - loss: 0,0663 Epoch 18/40 - 18s 66ms/step - loss: 0,0771 Epoch 19/40 - 16s 59ms/step - loss: 0,0454 Epoch 20/40 - 16s 61ms/step - loss: 0,0437 Epoch 21/40 - 17s 62ms/step - loss: 0,0460 Epoch 22/40 - 16s 59ms/step - loss: 0,0640 Epoch 23/40 - 15s 56ms/step - loss: 0,0662 Epoch 24/40 - 15s 56ms/step - loss: 0,0421 Epoch 25/40 - 16s 58ms/step - loss: 0,0431 Epoch 26/40 - 16s 57ms/step - loss: 0,0442 Epoch 27/40 - 16s 59ms/step - loss: 0,0421 Epoch 28/40 - 16s 57ms/step - loss: 0,0384 Epoch 29/40 - 15s 56ms/step - loss: 0,0402 Epoch 30/40 - 15s 56ms/step - loss: 0,0525 Timp (sec) 502,256733099956																																	
<table border="1"> <thead> <tr> <th colspan="5">Antrenare 80%</th> </tr> <tr> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> </tr> </thead> <tbody> <tr> <td>0,033</td> <td>-0,038</td> <td>0,133</td> <td>-0,008</td> <td>0,195</td> </tr> </tbody> </table>		Antrenare 80%					MAPE	ME	MAE	MPE	RMSE	0,033	-0,038	0,133	-0,008	0,195	<table border="1"> <thead> <tr> <th colspan="5">Testare 20%</th> </tr> <tr> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> </tr> </thead> <tbody> <tr> <td>0,062</td> <td>-0,145</td> <td>0,287</td> <td>-0,031</td> <td>0,383</td> </tr> </tbody> </table>		Testare 20%					MAPE	ME	MAE	MPE	RMSE	0,062	-0,145	0,287	-0,031	0,383
Antrenare 80%																																	
MAPE	ME	MAE	MPE	RMSE																													
0,033	-0,038	0,133	-0,008	0,195																													
Testare 20%																																	
MAPE	ME	MAE	MPE	RMSE																													
0,062	-0,145	0,287	-0,031	0,383																													

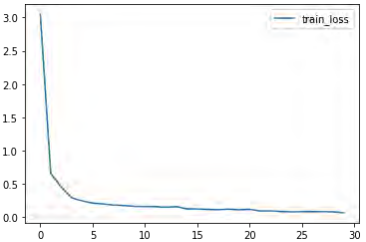
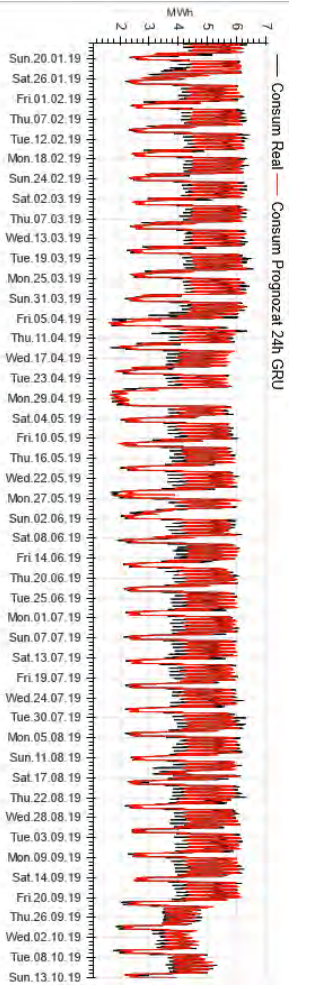
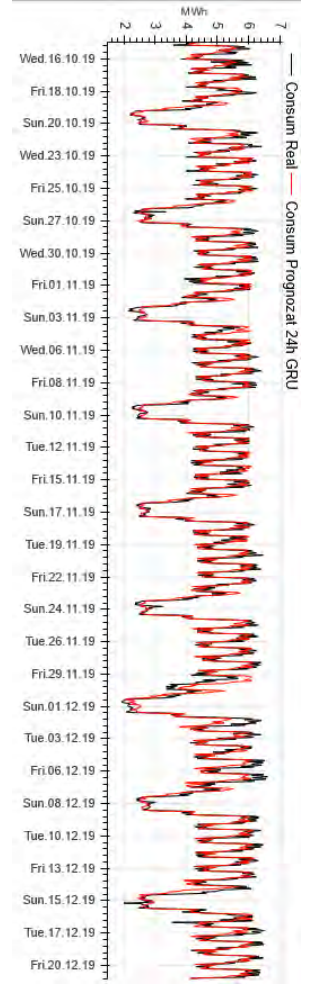


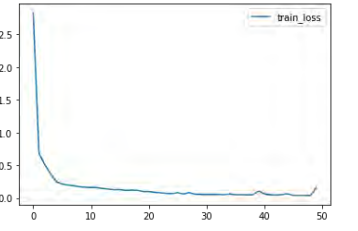
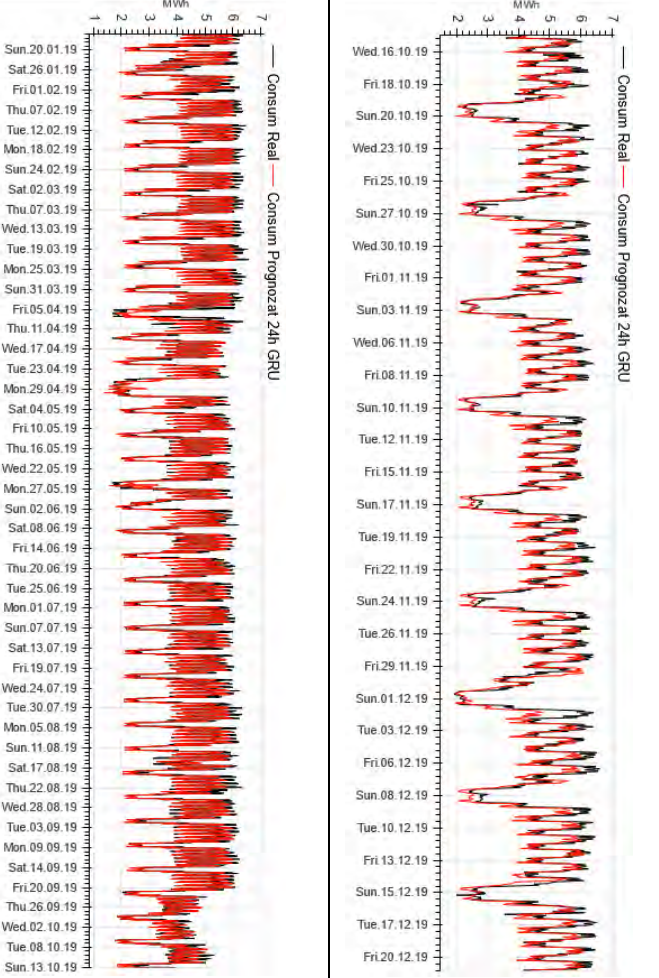




GRU_LC_agg_24h {GRU 24 3 240 240 168}	Factori exogeni								
verbose, epochs, batch_size = 1, 200, 24 model_gru = Sequential() model_gru.add(GRU(240, return_sequences=True,input_shape=(train_x.shape[1], train_x.shape[2]))) model_gru.add(GRU(240, return_sequences=True)) model_gru.add(GRU(168, return_sequences=False)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam')	Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14								
Epoch 1/200 - 26s 76ms/step - loss: 2,6542 Epoch 2/200 - 19s 69ms/step - loss: 0,5061 Epoch 3/200 - 18s 68ms/step - loss: 0,2676 Epoch 4/200 - 17s 62ms/step - loss: 0,2219 Epoch 5/200 - 18s 66ms/step - loss: 0,1789 Epoch 6/200 - 16s 60ms/step - loss: 0,1638 Epoch 7/200 - 18s 67ms/step - loss: 0,1431 Epoch 8/200 - 18s 65ms/step - loss: 0,1236 Epoch 9/200 - 18s 66ms/step - loss: 0,1227 Epoch 10/200 - 18s 67ms/step - loss: 0,0977 Epoch 11/200 - 19s 69ms/step - loss: 0,0931 Epoch 12/200 - 17s 63ms/step - loss: 0,1189 Epoch 13/200 - 18s 65ms/step - loss: 0,0865 Epoch 14/200 - 17s 63ms/step - loss: 0,1047 Epoch 15/200 - 18s 68ms/step - loss: 0,0731 Epoch 16/200 - 18s 67ms/step - loss: 0,0817 Epoch 17/200 - 16s 58ms/step - loss: 0,0708 Epoch 18/200 - 16s 60ms/step - loss: 0,0869 Epoch 19/200 - 17s 64ms/step - loss: 0,0713 Epoch 20/200 - 17s 63ms/step - loss: 0,0567 Epoch 21/200 - 19s 70ms/step - loss: 0,0467 Epoch 22/200 - 16s 58ms/step - loss: 0,0472 Epoch 23/200 - 16s 59ms/step - loss: 0,0443 Epoch 24/200 - 16s 57ms/step - loss: 0,0429 Epoch 25/200 - 15s 57ms/step - loss: 0,0544 Epoch 26/200 - 17s 61ms/step - loss: 0,0471 Epoch 27/200 - 17s 63ms/step - loss: 0,0513 Epoch 28/200 - 17s 62ms/step - loss: 0,0393 Epoch 29/200 - 16s 58ms/step - loss: 0,0455 Epoch 30/200 - 16s 61ms/step - loss: 0,0379 Epoch 31/200 - 16s 60ms/step - loss: 0,0386 Epoch 32/200 - 17s 63ms/step - loss: 0,0386 Epoch 33/200 - 20s 73ms/step - loss: 0,0371 Epoch 34/200 - 18s 66ms/step - loss: 0,0870 Epoch 35/200 - 26s 94ms/step - loss: 0,0354 Epoch 36/200 - 19s 72ms/step - loss: 0,0445 Epoch 37/200 - 21s 76ms/step - loss: 0,0324 Epoch 38/200 - 17s 64ms/step - loss: 0,0339 Epoch 39/200 - 26s 96ms/step - loss: 0,0604 Epoch 40/200 - 19s 69ms/step - loss: 0,0358 Epoch 185/200 - 17s 64ms/step - loss: 0,0047 Epoch 186/200 - 17s 64ms/step - loss: 0,0053 Epoch 187/200 - 18s 67ms/step - loss: 0,0054 Epoch 188/200 - 17s 63ms/step - loss: 0,0046 Epoch 189/200 - 17s 64ms/step - loss: 0,0044 Epoch 190/200 - 17s 62ms/step - loss: 0,0069 Epoch 191/200 - 18s 65ms/step - loss: 0,0041 Epoch 192/200 - 17s 64ms/step - loss: 0,0047 Epoch 193/200 - 17s 63ms/step - loss: 0,0039 Epoch 194/200 - 17s 64ms/step - loss: 0,0284 Epoch 195/200 - 17s 64ms/step - loss: 0,0206 Epoch 196/200 - 17s 63ms/step - loss: 0,0082 Epoch 197/200 - 17s 63ms/step - loss: 0,0057 Epoch 198/200 - 18s 65ms/step - loss: 0,0049 Epoch 199/200 - 17s 64ms/step - loss: 0,0043 Epoch 200/200 - 17s 64ms/step - loss: 0,0039 Timp (sec) 3535,2412583827972									
Antrenare 80%		Testare 20%							
MAPE	ME	MAE	MPE	RMS E	MAPE	ME	MAE	MPE	RMSE
0,0117	0,0007	0,047	0,001	0,06	0,057	-0,117	0,269	-0,021	0,366

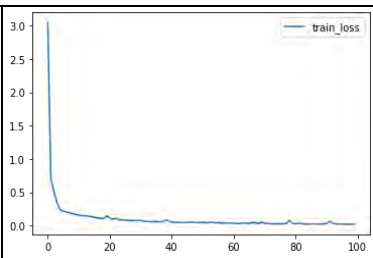


GRU_LC_agg_24h {GRU[24 3 80 80 48]}	Factori exogeni																															
<pre> verbose, epochs, batch_size = 1, 30, 24 model_gru = Sequential() model_gru.add(GRU(240, return_sequences=True,input_shape=(train_x.shape[1], train_x.shape[2]))) model_gru.add(GRU(240, return_sequences=True)) model_gru.add(GRU(168, return_sequences=False)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam') </pre>	<p>Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14</p>																															
<pre> Epoch 1/30 - 11s 19ms/step - loss: 6,3350 Epoch 2/30 - 6s 21ms/step - loss: 0,7597 Epoch 3/30 - 6s 21ms/step - loss: 0,5270 Epoch 4/30 - 6s 20ms/step - loss: 0,3159 Epoch 5/30 - 6s 21ms/step - loss: 0,2614 Epoch 6/30 - 6s 20ms/step - loss: 0,2211 Epoch 7/30 - 5s 20ms/step - loss: 0,1933 Epoch 8/30 - 5s 20ms/step - loss: 0,1974 Epoch 9/30 - 5s 20ms/step - loss: 0,1698 Epoch 10/30 - 5s 19ms/step - loss: 0,1673 Epoch 11/30 - 5s 20ms/step - loss: 0,1657 Epoch 12/30 - 5s 20ms/step - loss: 0,1571 Epoch 13/30 - 5s 20ms/step - loss: 0,1621 Epoch 14/30 - 6s 21ms/step - loss: 0,1814 Epoch 15/30 - 6s 20ms/step - loss: 0,1238 Epoch 16/30 - 6s 21ms/step - loss: 0,1252 Epoch 17/30 - 5s 20ms/step - loss: 0,1185 Epoch 18/30 - 5s 20ms/step - loss: 0,1206 Epoch 19/30 - 5s 20ms/step - loss: 0,1151 Epoch 20/30 - 5s 19ms/step - loss: 0,1186 Epoch 21/30 - 5s 19ms/step - loss: 0,1392 Epoch 22/30 - 5s 20ms/step - loss: 0,0958 Epoch 23/30 - 6s 20ms/step - loss: 0,0913 Epoch 24/30 - 5s 20ms/step - loss: 0,0903 Epoch 25/30 - 5s 19ms/step - loss: 0,0797 Epoch 26/30 - 5s 19ms/step - loss: 0,0897 Epoch 27/30 - 5s 19ms/step - loss: 0,0791 Epoch 28/30 - 5s 19ms/step - loss: 0,0838 Epoch 29/30 - 5s 20ms/step - loss: 0,0929 Epoch 30/30 - 5s 20ms/step - loss: 0,0624 Timp (sec) 167,48006868362427 </pre>																																
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="5">Antrenare 80%</th> <th colspan="5">Testare 20%</th> </tr> <tr> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> </tr> </thead> <tbody> <tr> <td>0,050</td> <td>0,03</td> <td>0,1988</td> <td>0,0148</td> <td>0,2724</td> <td>0,0609</td> <td>-0,081</td> <td>0,2855</td> <td>-0,0114</td> <td>0,3711</td> </tr> </tbody> </table>		Antrenare 80%					Testare 20%					MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE	0,050	0,03	0,1988	0,0148	0,2724	0,0609	-0,081	0,2855	-0,0114	0,3711
Antrenare 80%					Testare 20%																											
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE																							
0,050	0,03	0,1988	0,0148	0,2724	0,0609	-0,081	0,2855	-0,0114	0,3711																							

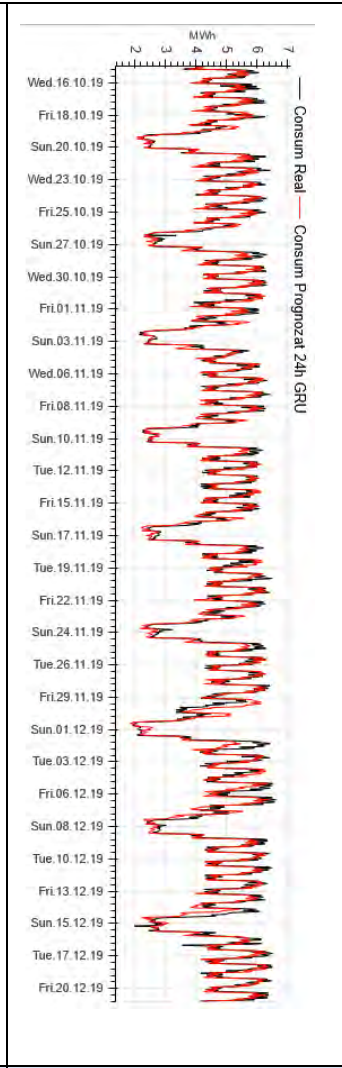
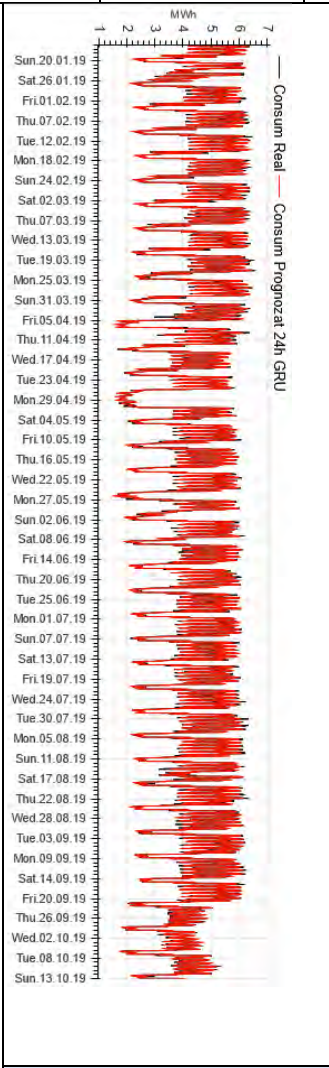
GRU_LC_agg_24h {GRU 24 3 80 80 48}	Factori exogeni								
<pre> verbose, epochs, batch_size = 1, 50, 24 model_gru = Sequential() model_gru.add(GRU(240, return_sequences=True,input_shape=(train_x.shape[1], train_x.shape[2]))) model_gru.add(GRU(240, return_sequences=True)) model_gru.add(GRU(168, return_sequences=False)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam') Epoch 1/50 - 11s 19ms/step - loss: 5,9600 Epoch 2/50 - 6s 21ms/step - loss: 0,7773 Epoch 3/50 - 6s 20ms/step - loss: 0,5585 Epoch 4/50 - 6s 20ms/step - loss: 0,4241 Epoch 5/50 - 5s 20ms/step - loss: 0,2681 Epoch 6/50 - 5s 20ms/step - loss: 0,2141 Epoch 7/50 - 5s 20ms/step - loss: 0,2007 Epoch 8/50 - 5s 20ms/step - loss: 0,1843 Epoch 9/50 - 5s 20ms/step - loss: 0,1709 Epoch 10/50 - 6s 21ms/step - loss: 0,1696 Epoch 11/50 - 5s 20ms/step - loss: 0,1696 Epoch 12/50 - 5s 20ms/step - loss: 0,1528 Epoch 13/50 - 6s 21ms/step - loss: 0,1439 Epoch 14/50 - 5s 20ms/step - loss: 0,1375 Epoch 15/50 - 6s 21ms/step - loss: 0,1339 Epoch 16/50 - 6s 21ms/step - loss: 0,1351 Epoch 17/50 - 6s 22ms/step - loss: 0,1205 Epoch 18/50 - 5s 20ms/step - loss: 0,1187 Epoch 19/50 - 5s 20ms/step - loss: 0,1110 Epoch 20/50 - 5s 20ms/step - loss: 0,1019 Epoch 21/50 - 6s 21ms/step - loss: 0,1085 Epoch 22/50 - 6s 20ms/step - loss: 0,0858 Epoch 23/50 - 5s 20ms/step - loss: 0,0838 Epoch 24/50 - 6s 20ms/step - loss: 0,0719 Epoch 25/50 - 6s 21ms/step - loss: 0,0687 Epoch 26/50 - 6s 21ms/step - loss: 0,0950 Epoch 27/50 - 6s 21ms/step - loss: 0,0660 Epoch 28/50 - 6s 20ms/step - loss: 0,0799 Epoch 29/50 - 6s 21ms/step - loss: 0,0592 Epoch 30/50 - 5s 20ms/step - loss: 0,0581 Epoch 31/50 - 5s 20ms/step - loss: 0,0537 Epoch 32/50 - 6s 21ms/step - loss: 0,0535 Epoch 33/50 - 6s 21ms/step - loss: 0,0527 Epoch 34/50 - 6s 21ms/step - loss: 0,0515 Epoch 35/50 - 6s 21ms/step - loss: 0,0676 Epoch 36/50 - 6s 21ms/step - loss: 0,0530 Epoch 37/50 - 5s 20ms/step - loss: 0,0519 Epoch 38/50 - 6s 21ms/step - loss: 0,0518 Epoch 39/50 - 5s 20ms/step - loss: 0,0519 Epoch 40/50 - 5s 20ms/step - loss: 0,1016 Epoch 41/50 - 6s 21ms/step - loss: 0,0692 Epoch 42/50 - 5s 20ms/step - loss: 0,0577 Epoch 43/50 - 5s 20ms/step - loss: 0,0441 Epoch 44/50 - 5s 20ms/step - loss: 0,0493 Epoch 45/50 - 5s 20ms/step - loss: 0,0832 Epoch 46/50 - 6s 20ms/step - loss: 0,0428 Epoch 47/50 - 6s 22ms/step - loss: 0,0399 Epoch 48/50 - 6s 21ms/step - loss: 0,0447 Epoch 49/50 - 5s 20ms/step - loss: 0,0379 Epoch 50/50 - 5s 20ms/step - loss: 0,0470 Time (sec) 282,49980664253235 </pre>	<p>Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14</p> 								
Antrenare 80%		Testare 20%							
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0375	0,0207	0,147	0,008	0,2107	0,0597	0,0857	0,2749	0,0139	0,3555

```
GRU_IC_agg_24h {GRU[24|3|80|80|48]}
verbose, epochs, batch_size = 1, 100, 24
model_gru = Sequential()
model_gru.add(GRU(240,
return_sequences=True,input_shape=(train_x.shape[1], train_x.shape[2])))
model_gru.add(GRU(240, return_sequences=True))
model_gru.add(GRU(168, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
```

Factori exogeni
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14



Epoch 1/100 - 11s 22ms/step - loss: 6,4494
Epoch 2/100 - 6s 22ms/step - loss: 0,7885
Epoch 3/100 - 6s 22ms/step - loss: 0,5404
Epoch 4/100 - 6s 22ms/step - loss: 0,3885
Epoch 5/100 - 6s 22ms/step - loss: 0,2551
Epoch 6/100 - 6s 22ms/step - loss: 0,2171
Epoch 7/100 - 6s 23ms/step - loss: 0,2163
Epoch 8/100 - 6s 23ms/step - loss: 0,1933
Epoch 9/100 - 6s 22ms/step - loss: 0,1786
Epoch 10/100 - 6s 22ms/step - loss: 0,1706
Epoch 11/100 - 6s 22ms/step - loss: 0,1621
Epoch 12/100 - 6s 22ms/step - loss: 0,1410
Epoch 13/100 - 6s 22ms/step - loss: 0,1589
Epoch 14/100 - 7s 25ms/step - loss: 0,1309
Epoch 15/100 - 6s 23ms/step - loss: 0,1374
Epoch 16/100 - 6s 23ms/step - loss: 0,1268
Epoch 17/100 - 6s 23ms/step - loss: 0,1168
Epoch 18/100 - 6s 23ms/step - loss: 0,1099
Epoch 19/100 - 6s 22ms/step - loss: 0,1073
Epoch 20/100 - 6s 24ms/step - loss: 0,1505
Epoch 21/100 - 6s 22ms/step - loss: 0,1272
Epoch 22/100 - 6s 23ms/step - loss: 0,0939
Epoch 23/100 - 6s 23ms/step - loss: 0,1247
Epoch 24/100 - 6s 23ms/step - loss: 0,0869
Epoch 25/100 - 6s 22ms/step - loss: 0,0859
Epoch 26/100 - 6s 23ms/step - loss: 0,0775
Epoch 27/100 - 6s 22ms/step - loss: 0,0842
Epoch 28/100 - 6s 22ms/step - loss: 0,0749
Epoch 29/100 - 6s 23ms/step - loss: 0,0781
Epoch 30/100 - 6s 22ms/step - loss: 0,0885
Epoch 31/100 - 6s 23ms/step - loss: 0,1002
Epoch 32/100 - 6s 22ms/step - loss: 0,0616
Epoch 33/100 - 6s 22ms/step - loss: 0,0614
Epoch 34/100 - 6s 23ms/step - loss: 0,0583
Epoch 35/100 - 6s 23ms/step - loss: 0,0557
Epoch 36/100 - 6s 23ms/step - loss: 0,0627
Epoch 37/100 - 6s 22ms/step - loss: 0,0559
Epoch 38/100 - 6s 22ms/step - loss: 0,0537
Epoch 39/100 - 6s 23ms/step - loss: 0,0561
Epoch 40/100 - 6s 22ms/step - loss: 0,0886
Epoch 41/100 - 6s 23ms/step - loss: 0,0533
Epoch 42/100 - 6s 23ms/step - loss: 0,0493
Epoch 43/100 - 6s 21ms/step - loss: 0,0476
Epoch 44/100 - 6s 22ms/step - loss: 0,0478
Epoch 45/100 - 6s 23ms/step - loss: 0,0454
Epoch 46/100 - 7s 25ms/step - loss: 0,0460
Epoch 47/100 - 6s 22ms/step - loss: 0,0468
Epoch 48/100 - 6s 22ms/step - loss: 0,0517
Epoch 49/100 - 6s 22ms/step - loss: 0,0406
Epoch 50/100 - 6s 22ms/step - loss: 0,0448
Epoch 51/100 - 6s 23ms/step - loss: 0,0511
Epoch 52/100 - 6s 23ms/step - loss: 0,0414
Epoch 53/100 - 6s 23ms/step - loss: 0,0422
Epoch 54/100 - 6s 22ms/step - loss: 0,0536
Epoch 55/100 - 6s 23ms/step - loss: 0,0379
.....
Epoch 95/100 - 6s 23ms/step - loss: 0,0263
Epoch 96/100 - 6s 23ms/step - loss: 0,0212
Epoch 97/100 - 6s 23ms/step - loss: 0,0201
Epoch 98/100 - 6s 22ms/step - loss: 0,0188
Epoch 99/100 - 6s 22ms/step - loss: 0,0195
Epoch 100/100 - 6s 23ms/step - loss: 0,0200
Time (sec) 613,240161895752



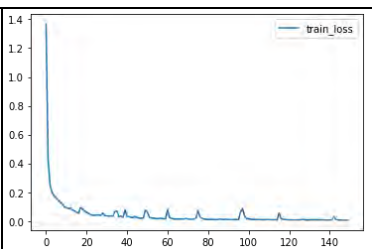
Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,027	-0,0032	0,11	0,001	0,1462	0,055	-0,096	0,259	-0,018	0,348

```

GRU_IC_agg_24h {GRU[24|3|80|80|48]}
verbose, epochs, batch_size = 1, 150, 24
model_gru = Sequential()
model_gru.add(GRU(240,
return_sequences=True,input_shape=(train_x.shape
[1], train_x.shape[2])))
model_gru.add(GRU(240, return_sequences=True))
model_gru.add(GRU(168, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')

```

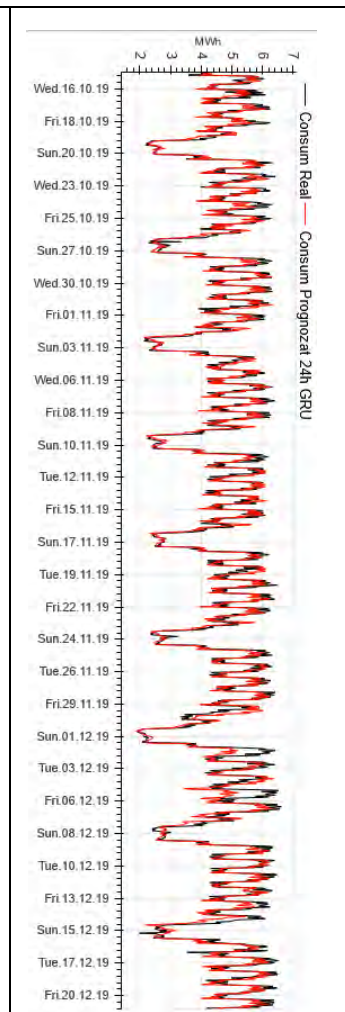
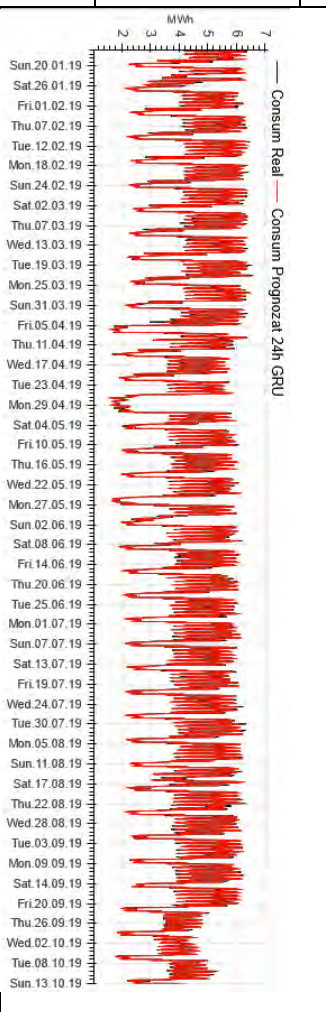
Factori exogeni
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14



```

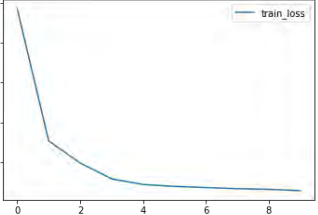
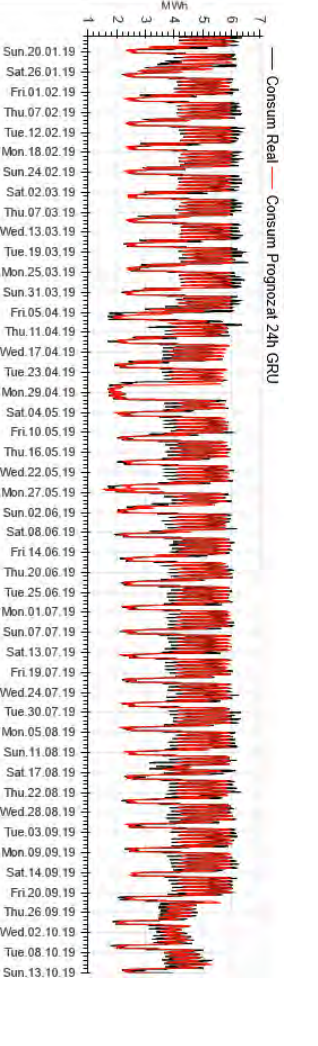
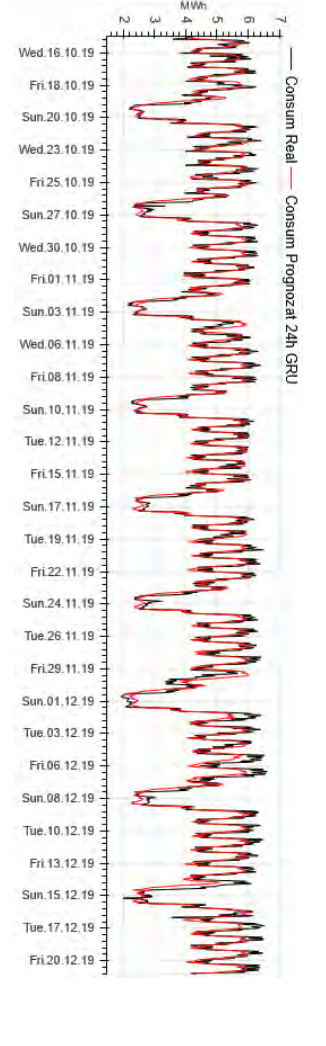
Epoch 1/200 - 11s 20ms/step - loss: 6,7085
Epoch 2/200 - 6s 20ms/step - loss: 1,3728
Epoch 3/200 - 5s 18ms/step - loss: 0,5984
Epoch 4/200 - 5s 20ms/step - loss: 0,4337
Epoch 5/200 - 5s 20ms/step - loss: 0,2757
Epoch 6/200 - 5s 19ms/step - loss: 0,2412
Epoch 7/200 - 5s 19ms/step - loss: 0,2128
Epoch 8/200 - 5s 20ms/step - loss: 0,1885
Epoch 9/200 - 5s 20ms/step - loss: 0,1755
Epoch 10/200 - 7s 26ms/step - loss: 0,1745
Epoch 11/200 - 5s 19ms/step - loss: 0,1638
Epoch 12/200 - 5s 18ms/step - loss: 0,1813
Epoch 13/200 - 5s 18ms/step - loss: 0,1372
Epoch 14/200 - 6s 21ms/step - loss: 0,1310
Epoch 15/200 - 6s 21ms/step - loss: 0,1450
Epoch 16/200 - 5s 19ms/step - loss: 0,1365
Epoch 17/200 - 5s 19ms/step - loss: 0,1396
Epoch 18/200 - 7s 26ms/step - loss: 0,1128
Epoch 19/200 - 5s 20ms/step - loss: 0,1022
Epoch 20/200 - 6s 20ms/step - loss: 0,1075
Epoch 21/200 - 5s 20ms/step - loss: 0,0910
Epoch 22/200 - 5s 20ms/step - loss: 0,0902
Epoch 23/200 - 5s 19ms/step - loss: 0,0853
Epoch 24/200 - 5s 20ms/step - loss: 0,1564
Epoch 25/200 - 6s 21ms/step - loss: 0,1046
Epoch 26/200 - 5s 20ms/step - loss: 0,1101
Epoch 27/200 - 6s 21ms/step - loss: 0,0728
Epoch 28/200 - 7s 25ms/step - loss: 0,0662
Epoch 29/200 - 6s 21ms/step - loss: 0,0641
Epoch 30/200 - 6s 21ms/step - loss: 0,0617
Epoch 31/200 - 5s 19ms/step - loss: 0,0709
Epoch 32/200 - 5s 19ms/step - loss: 0,0681
Epoch 33/200 - 5s 19ms/step - loss: 0,0739
Epoch 34/200 - 5s 19ms/step - loss: 0,0550
Epoch 35/200 - 5s 18ms/step - loss: 0,0575
Epoch 36/200 - 5s 20ms/step - loss: 0,0544
Epoch 37/200 - 5s 19ms/step - loss: 0,0677
Epoch 38/200 - 5s 20ms/step - loss: 0,0498
Epoch 39/200 - 7s 28ms/step - loss: 0,0626
Epoch 40/200 - 6s 21ms/step - loss: 0,0674
.....
Epoch 140/150 - 10s 36ms/step - loss: 0,0114
Epoch 141/150 - 9s 34ms/step - loss: 0,0555
Epoch 142/150 - 9s 32ms/step - loss: 0,0170
Epoch 143/150 - 9s 34ms/step - loss: 0,0116
Epoch 144/150 - 9s 33ms/step - loss: 0,0105
Epoch 145/150 - 9s 34ms/step - loss: 0,0096
Epoch 146/150 - 10s 36ms/step - loss: 0,0090
Epoch 147/150 - 10s 36ms/step - loss: 0,0093
Epoch 148/150 - 10s 35ms/step - loss: 0,0090
Epoch 149/150 - 10s 39ms/step - loss: 0,0090
Epoch 150/150 - 10s 37ms/step - loss: 0,0094
Timp (sec) 1411,458660364151

```

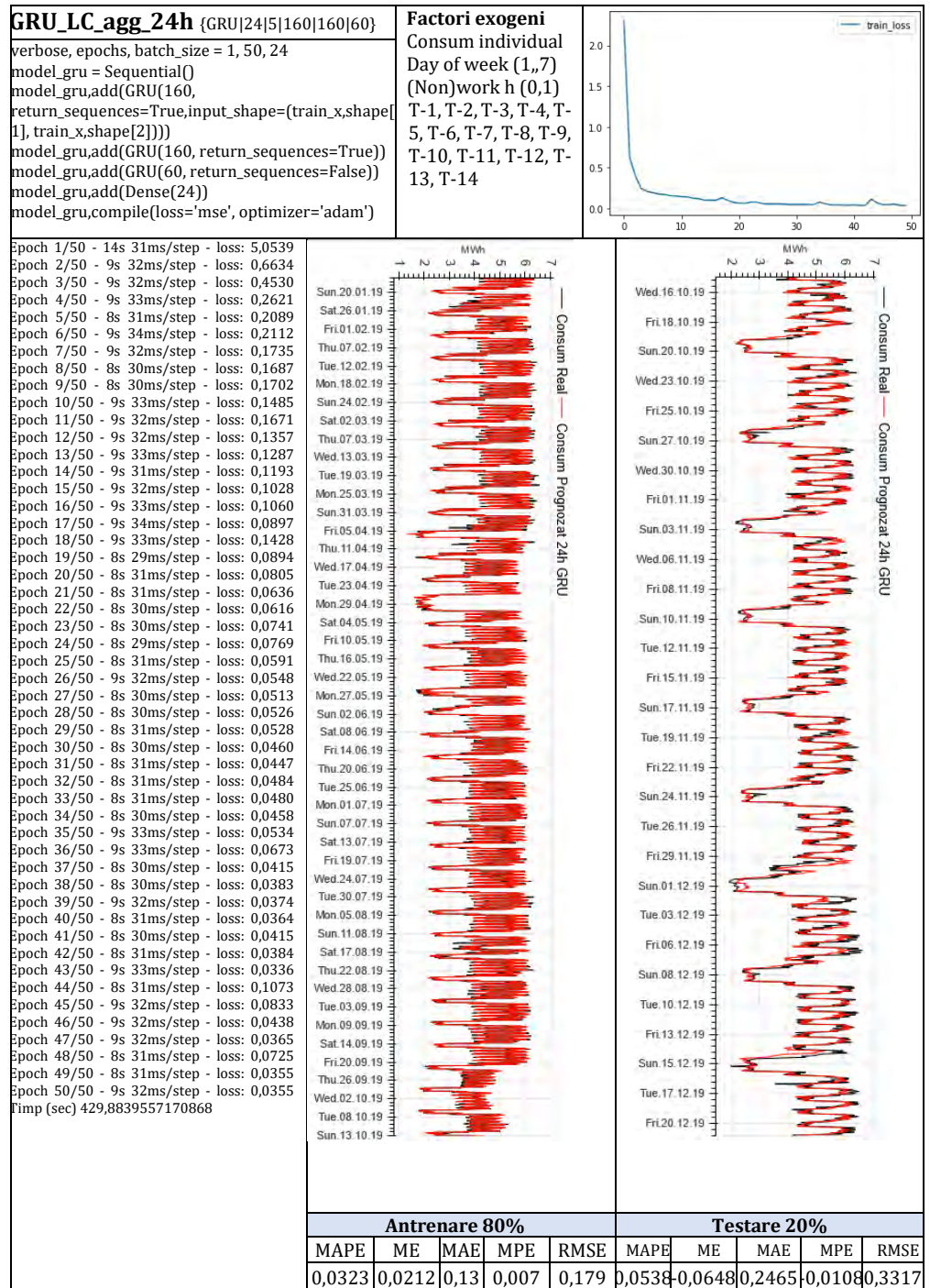


Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,021	-0,0095	0,0815	-0,0023	0,1049	0,0587	-0,1128	0,2851	-0,0189	0,3833

GRU_LC_agg_24h {GRU 24 3 80 80 48}	Factori exogeni Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14									
<pre> verbose, epochs, batch_size = 1, 200, 24 model_gru = Sequential() model_gru.add(GRU(240, return_sequences=True,input_shape=(train_x.shape[1], train_x.shape[2]))) model_gru.add(GRU(240, return_sequences=True)) model_gru.add(GRU(168, return_sequences=False)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam') </pre>										
<pre> Epoch 1/200 - 11s 20ms/step - loss: 6,7085 Epoch 2/200 - 6s 20ms/step - loss: 1,3728 Epoch 3/200 - 5s 18ms/step - loss: 0,5984 Epoch 4/200 - 5s 20ms/step - loss: 0,4337 Epoch 5/200 - 5s 20ms/step - loss: 0,2757 Epoch 6/200 - 5s 19ms/step - loss: 0,2412 Epoch 7/200 - 5s 19ms/step - loss: 0,2128 Epoch 8/200 - 5s 20ms/step - loss: 0,1885 Epoch 9/200 - 5s 20ms/step - loss: 0,1755 Epoch 10/200 - 7s 26ms/step - loss: 0,1745 Epoch 11/200 - 5s 19ms/step - loss: 0,1638 Epoch 12/200 - 5s 18ms/step - loss: 0,1813 Epoch 13/200 - 5s 18ms/step - loss: 0,1372 Epoch 14/200 - 6s 21ms/step - loss: 0,1310 Epoch 15/200 - 6s 21ms/step - loss: 0,1450 Epoch 16/200 - 5s 19ms/step - loss: 0,1365 Epoch 17/200 - 5s 19ms/step - loss: 0,1396 Epoch 18/200 - 7s 26ms/step - loss: 0,1128 Epoch 19/200 - 5s 20ms/step - loss: 0,1022 Epoch 20/200 - 6s 20ms/step - loss: 0,1075 Epoch 21/200 - 5s 20ms/step - loss: 0,0910 Epoch 22/200 - 5s 20ms/step - loss: 0,0902 Epoch 23/200 - 5s 19ms/step - loss: 0,0853 Epoch 24/200 - 5s 20ms/step - loss: 0,1564 Epoch 25/200 - 6s 21ms/step - loss: 0,1046 Epoch 26/200 - 5s 20ms/step - loss: 0,1101 Epoch 27/200 - 6s 21ms/step - loss: 0,0728 Epoch 28/200 - 7s 25ms/step - loss: 0,0662 Epoch 29/200 - 6s 21ms/step - loss: 0,0641 Epoch 30/200 - 6s 21ms/step - loss: 0,0617 Epoch 31/200 - 5s 19ms/step - loss: 0,0709 Epoch 32/200 - 5s 19ms/step - loss: 0,0681 Epoch 33/200 - 5s 19ms/step - loss: 0,0739 Epoch 34/200 - 5s 19ms/step - loss: 0,0550 Epoch 35/200 - 5s 18ms/step - loss: 0,0575 Epoch 36/200 - 5s 20ms/step - loss: 0,0544 Epoch 37/200 - 5s 19ms/step - loss: 0,0677 Epoch 38/200 - 5s 20ms/step - loss: 0,0498 Epoch 39/200 - 7s 28ms/step - loss: 0,0626 Epoch 40/200 - 6s 21ms/step - loss: 0,0674 Epoch 180/200 - 5s 19ms/step - loss: 0,0125 Epoch 181/200 - 6s 23ms/step - loss: 0,0134 Epoch 182/200 - 5s 20ms/step - loss: 0,0133 Epoch 183/200 - 5s 19ms/step - loss: 0,0124 Epoch 184/200 - 5s 19ms/step - loss: 0,0138 Epoch 185/200 - 5s 18ms/step - loss: 0,0126 Epoch 186/200 - 5s 19ms/step - loss: 0,0127 Epoch 187/200 - 5s 19ms/step - loss: 0,0120 Epoch 188/200 - 5s 19ms/step - loss: 0,0142 Epoch 189/200 - 6s 21ms/step - loss: 0,0117 Epoch 190/200 - 5s 19ms/step - loss: 0,0115 Epoch 191/200 - 5s 19ms/step - loss: 0,0261 Epoch 192/200 - 5s 19ms/step - loss: 0,0418 Epoch 193/200 - 5s 19ms/step - loss: 0,0156 Epoch 194/200 - 5s 19ms/step - loss: 0,0127 Epoch 195/200 - 5s 18ms/step - loss: 0,0119 Epoch 196/200 - 5s 19ms/step - loss: 0,0111 Epoch 197/200 - 5s 19ms/step - loss: 0,0107 Epoch 198/200 - 5s 19ms/step - loss: 0,0112 Epoch 199/200 - 5s 19ms/step - loss: 0,0106 Epoch 200/200 - 5s 19ms/step - loss: 0,0107 </pre>										
	Antrenare 80%				Testare 20%					
	MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
	0,0199	-0,0095	0,0815	-0,0023	0,1049	0,0699	-0,1128	0,2851	-0,0189	0,3833
Time (sec) 1099,8107426166534										

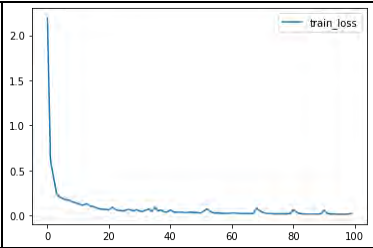
GRU_LC_agg_24h {GRU 24 5 160 160 60}	Factori exogeni								
<pre> verbose, epochs, batch_size = 1, 10, 24 model_gru = Sequential() model_gru.add(GRU(160, return_sequences=True,input_shape=(train_x,shape 1), train_x,shape[2]))) model_gru.add(GRU(160, return_sequences=True)) model_gru.add(GRU(60, return_sequences=False)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam') </pre>	<p>Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14</p>								
<pre> Epoch 1/10 - 14s 33ms/step - loss: 4,7638 Epoch 2/10 - 9s 33ms/step - loss: 0,9095 Epoch 3/10 - 9s 35ms/step - loss: 0,5627 Epoch 4/10 - 9s 34ms/step - loss: 0,3299 Epoch 5/10 - 9s 33ms/step - loss: 0,2235 Epoch 6/10 - 9s 33ms/step - loss: 0,1911 Epoch 7/10 - 10s 36ms/step - loss: 0,1826 Epoch 8/10 - 10s 36ms/step - loss: 0,1661 Epoch 9/10 - 10s 37ms/step - loss: 0,1680 Epoch 10/10 - 10s 35ms/step - loss: 0,1449 Timp (sec) 99,00997948646545 </pre>									
Antrenare 80%		Testare 20%							
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,072	0,013	0,280	0,0136	0,401	0,071	-0,1418	0,3286	-0,026	0,4242

GRU_LC_agg_24h {GRU 24 5 160 160 60} verbose, epochs, batch_size = 1, 30, 24 model_gru = Sequential() model_gru.add(GRU(160, return_sequences=True,input_shape=(train_x.shape[1], train_x.shape[2]))) model_gru.add(GRU(160, return_sequences=True)) model_gru.add(GRU(60, return_sequences=False)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam')	Factori exogeni Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14																																
Epoch 1/30 - 13s 30ms/step - loss: 4,9577 Epoch 2/30 - 8s 30ms/step - loss: 1,0366 Epoch 3/30 - 8s 30ms/step - loss: 0,5280 Epoch 4/30 - 8s 31ms/step - loss: 0,3324 Epoch 5/30 - 8s 30ms/step - loss: 0,2460 Epoch 6/30 - 8s 30ms/step - loss: 0,1978 Epoch 7/30 - 8s 31ms/step - loss: 0,1858 Epoch 8/30 - 10s 35ms/step - loss: 0,1770 Epoch 9/30 - 9s 32ms/step - loss: 0,1651 Epoch 10/30 - 10s 36ms/step - loss: 0,1598 Epoch 11/30 - 9s 33ms/step - loss: 0,1302 Epoch 12/30 - 8s 30ms/step - loss: 0,1326 Epoch 13/30 - 9s 33ms/step - loss: 0,1182 Epoch 14/30 - 9s 33ms/step - loss: 0,1133 Epoch 15/30 - 9s 33ms/step - loss: 0,1111 Epoch 16/30 - 9s 34ms/step - loss: 0,0925 Epoch 17/30 - 9s 33ms/step - loss: 0,0965 Epoch 18/30 - 9s 33ms/step - loss: 0,1103 Epoch 19/30 - 9s 33ms/step - loss: 0,0911 Epoch 20/30 - 8s 30ms/step - loss: 0,0842 Epoch 21/30 - 9s 33ms/step - loss: 0,0638 Epoch 22/30 - 8s 30ms/step - loss: 0,0606 Epoch 23/30 - 8s 31ms/step - loss: 0,0584 Epoch 24/30 - 9s 32ms/step - loss: 0,0564 Epoch 25/30 - 9s 32ms/step - loss: 0,0555 Epoch 26/30 - 9s 32ms/step - loss: 0,0897 Epoch 27/30 - 8s 31ms/step - loss: 0,0601 Epoch 28/30 - 9s 31ms/step - loss: 0,0544 Epoch 29/30 - 8s 31ms/step - loss: 0,0493 Epoch 30/30 - 10s 36ms/step - loss: 0,0505 Timp (sec) 265,5710458755493																																	
		<table border="1"> <thead> <tr> <th colspan="5">Antrenare 80%</th> </tr> <tr> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> </tr> </thead> <tbody> <tr> <td>0,0595</td> <td>0,178</td> <td>0,218</td> <td>0,0508</td> <td>0,3057</td> </tr> </tbody> </table>	Antrenare 80%					MAPE	ME	MAE	MPE	RMSE	0,0595	0,178	0,218	0,0508	0,3057	<table border="1"> <thead> <tr> <th colspan="5">Testare 20%</th> </tr> <tr> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> </tr> </thead> <tbody> <tr> <td>0,0622</td> <td>0,084</td> <td>0,27</td> <td>0,026</td> <td>0,3544</td> </tr> </tbody> </table>	Testare 20%					MAPE	ME	MAE	MPE	RMSE	0,0622	0,084	0,27	0,026	0,3544
Antrenare 80%																																	
MAPE	ME	MAE	MPE	RMSE																													
0,0595	0,178	0,218	0,0508	0,3057																													
Testare 20%																																	
MAPE	ME	MAE	MPE	RMSE																													
0,0622	0,084	0,27	0,026	0,3544																													

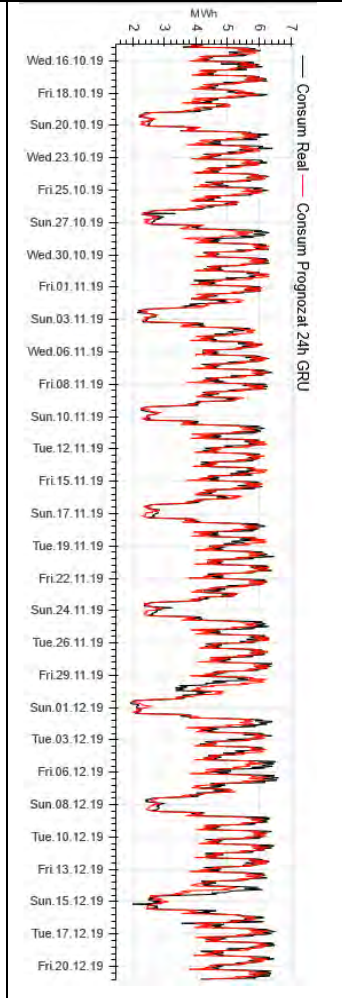
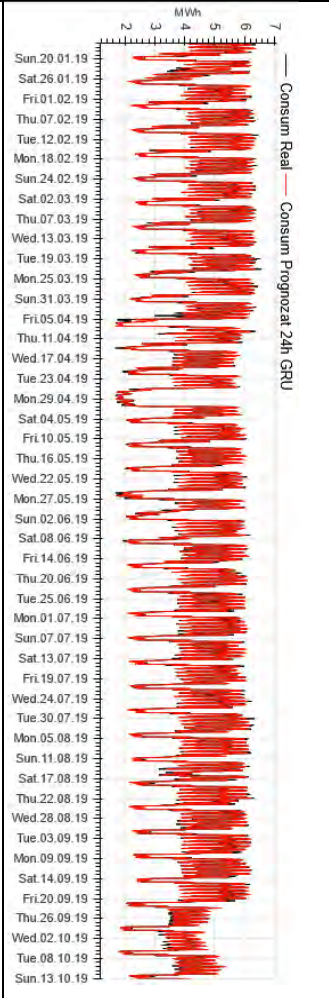



```
GRU_LC_agg_24h {GRU|24|5|160|160|60}
verbose, epochs, batch_size = 1, 100, 24
model_gru = Sequential()
model_gru.add(GRU(160,
return_sequences=True,input_shape=(train_x.shape[1], train_x.shape[2])))
model_gru.add(GRU(160, return_sequences=True))
model_gru.add(GRU(60, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
```

Factori exogeni
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14



Epoch 1/100 - 11s 22ms/step - loss: 6,4494
Epoch 2/100 - 6s 22ms/step - loss: 0,7885
Epoch 3/100 - 6s 22ms/step - loss: 0,5404
Epoch 4/100 - 6s 22ms/step - loss: 0,3885
Epoch 5/100 - 6s 22ms/step - loss: 0,2551
Epoch 6/100 - 6s 22ms/step - loss: 0,2171
Epoch 7/100 - 6s 23ms/step - loss: 0,2163
Epoch 8/100 - 6s 23ms/step - loss: 0,1933
Epoch 9/100 - 6s 22ms/step - loss: 0,1786
Epoch 10/100 - 6s 22ms/step - loss: 0,1706
Epoch 11/100 - 6s 22ms/step - loss: 0,1621
Epoch 12/100 - 6s 22ms/step - loss: 0,1410
Epoch 13/100 - 6s 22ms/step - loss: 0,1589
Epoch 14/100 - 7s 25ms/step - loss: 0,1309
Epoch 15/100 - 6s 23ms/step - loss: 0,1374
Epoch 16/100 - 6s 23ms/step - loss: 0,1268
Epoch 17/100 - 6s 23ms/step - loss: 0,1168
Epoch 18/100 - 6s 23ms/step - loss: 0,1099
Epoch 19/100 - 6s 22ms/step - loss: 0,1073
Epoch 20/100 - 6s 24ms/step - loss: 0,1505
Epoch 21/100 - 6s 22ms/step - loss: 0,1272
Epoch 22/100 - 6s 23ms/step - loss: 0,0939
Epoch 23/100 - 6s 23ms/step - loss: 0,1247
Epoch 24/100 - 6s 23ms/step - loss: 0,0869
Epoch 25/100 - 6s 22ms/step - loss: 0,0859
Epoch 26/100 - 6s 23ms/step - loss: 0,0775
Epoch 27/100 - 6s 22ms/step - loss: 0,0842
Epoch 28/100 - 6s 22ms/step - loss: 0,0749
Epoch 29/100 - 6s 23ms/step - loss: 0,0781
Epoch 30/100 - 6s 22ms/step - loss: 0,0885
Epoch 31/100 - 6s 23ms/step - loss: 0,1002
Epoch 32/100 - 6s 22ms/step - loss: 0,0616
Epoch 33/100 - 6s 22ms/step - loss: 0,0614
Epoch 34/100 - 6s 23ms/step - loss: 0,0583
Epoch 35/100 - 6s 23ms/step - loss: 0,0557
Epoch 36/100 - 6s 23ms/step - loss: 0,0627
Epoch 37/100 - 6s 22ms/step - loss: 0,0559
Epoch 38/100 - 6s 22ms/step - loss: 0,0537
Epoch 39/100 - 6s 23ms/step - loss: 0,0561
Epoch 40/100 - 6s 22ms/step - loss: 0,0886
Epoch 41/100 - 6s 23ms/step - loss: 0,0533
Epoch 42/100 - 6s 23ms/step - loss: 0,0493
Epoch 43/100 - 6s 21ms/step - loss: 0,0476
Epoch 44/100 - 6s 22ms/step - loss: 0,0478
Epoch 45/100 - 6s 23ms/step - loss: 0,0454
Epoch 46/100 - 7s 25ms/step - loss: 0,0460
Epoch 47/100 - 6s 22ms/step - loss: 0,0468
Epoch 48/100 - 6s 22ms/step - loss: 0,0517
Epoch 49/100 - 6s 22ms/step - loss: 0,0406
Epoch 50/100 - 6s 22ms/step - loss: 0,0448
Epoch 51/100 - 6s 23ms/step - loss: 0,0511
Epoch 52/100 - 6s 23ms/step - loss: 0,0414
Epoch 53/100 - 6s 23ms/step - loss: 0,0422



.....
Epoch 95/100 - 6s 23ms/step - loss: 0,0263
Epoch 96/100 - 6s 23ms/step - loss: 0,0212
Epoch 97/100 - 6s 23ms/step - loss: 0,0201
Epoch 98/100 - 6s 22ms/step - loss: 0,0188
Epoch 99/100 - 6s 22ms/step - loss: 0,0195
Epoch 100/100 - 6s 23ms/step - loss: 0,0200
Timp (sec) 613,240161895752

Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0277	0,005	0,110	0,0035	0,1554	0,057	-0,0994	0,268	-0,018	0,351

GRU_LC_agg_24h {GRU|24|5|160|160|60}

```

verbose, epochs, batch_size = 1, 150, 24
model_gru = Sequential()
model_gru.add(GRU(160,
return_sequences=True,input_shape=(train_x.shape[1],
train_x.shape[2])))
model_gru.add(GRU(160, return_sequences=True))
model_gru.add(GRU(60, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')

```

Factori exogeni

Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14

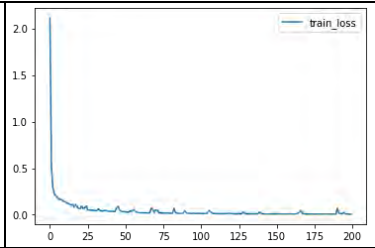
Epoch 1/150	- 16s	38ms/step	- loss: 4,9235
Epoch 2/150	- 9s	33ms/step	- loss: 0,6873
Epoch 3/150	- 9s	33ms/step	- loss: 0,4397
Epoch 4/150	- 11s	40ms/step	- loss: 0,2862
Epoch 5/150	- 11s	41ms/step	- loss: 0,2124
Epoch 6/150	- 10s	36ms/step	- loss: 0,1948
Epoch 7/150	- 10s	35ms/step	- loss: 0,1886
Epoch 8/150	- 10s	35ms/step	- loss: 0,1755
Epoch 9/150	- 9s	35ms/step	- loss: 0,1549
Epoch 10/150	- 9s	34ms/step	- loss: 0,1382
Epoch 11/150	- 9s	34ms/step	- loss: 0,1399
Epoch 12/150	- 9s	33ms/step	- loss: 0,1341
Epoch 13/150	- 9s	34ms/step	- loss: 0,1438
Epoch 14/150	- 9s	33ms/step	- loss: 0,1267
Epoch 15/150	- 9s	32ms/step	- loss: 0,1305
Epoch 16/150	- 9s	35ms/step	- loss: 0,1173
Epoch 17/150	- 9s	33ms/step	- loss: 0,0888
Epoch 18/150	- 9s	32ms/step	- loss: 0,0850
Epoch 19/150	- 9s	33ms/step	- loss: 0,0807
Epoch 20/150	- 9s	34ms/step	- loss: 0,0740
Epoch 21/150	- 9s	34ms/step	- loss: 0,0842
Epoch 22/150	- 9s	34ms/step	- loss: 0,0816
Epoch 23/150	- 9s	33ms/step	- loss: 0,0632
Epoch 24/150	- 9s	34ms/step	- loss: 0,0905
Epoch 25/150	- 10s	36ms/step	- loss: 0,0526
Epoch 26/150	- 9s	33ms/step	- loss: 0,0479
Epoch 27/150	- 9s	33ms/step	- loss: 0,0481
Epoch 28/150	- 9s	33ms/step	- loss: 0,0453
Epoch 29/150	- 10s	36ms/step	- loss: 0,0530
Epoch 30/150	- 9s	34ms/step	- loss: 0,0801
Epoch 31/150	- 9s	33ms/step	- loss: 0,0433
Epoch 32/150	- 9s	32ms/step	- loss: 0,0422
Epoch 33/150	- 9s	33ms/step	- loss: 0,0393
Epoch 34/150	- 9s	33ms/step	- loss: 0,0425
Epoch 35/150	- 9s	34ms/step	- loss: 0,0482
Epoch 36/150	- 9s	35ms/step	- loss: 0,0398
Epoch 37/150	- 9s	35ms/step	- loss: 0,0396
Epoch 38/150	- 9s	33ms/step	- loss: 0,0528
Epoch 39/150	- 9s	35ms/step	- loss: 0,0784
Epoch 40/150	- 9s	35ms/step	- loss: 0,0620
.....			
Epoch 133/150	- 9s	33ms/step	- loss: 0,0107
Epoch 134/150	- 9s	33ms/step	- loss: 0,0115
Epoch 135/150	- 10s	36ms/step	- loss: 0,0121
Epoch 136/150	- 11s	40ms/step	- loss: 0,0111
Epoch 137/150	- 9s	35ms/step	- loss: 0,0106
Epoch 138/150	- 10s	35ms/step	- loss: 0,0108
Epoch 139/150	- 10s	36ms/step	- loss: 0,0138
Epoch 140/150	- 10s	36ms/step	- loss: 0,0114
Epoch 141/150	- 9s	34ms/step	- loss: 0,0555
Epoch 142/150	- 9s	32ms/step	- loss: 0,0170
Epoch 143/150	- 9s	34ms/step	- loss: 0,0116
Epoch 144/150	- 9s	33ms/step	- loss: 0,0105
Epoch 145/150	- 9s	34ms/step	- loss: 0,0096
Epoch 146/150	- 10s	36ms/step	- loss: 0,0090
Epoch 147/150	- 10s	36ms/step	- loss: 0,0093
Epoch 148/150	- 10s	35ms/step	- loss: 0,0090
Epoch 149/150	- 10s	39ms/step	- loss: 0,0090
Epoch 150/150	- 10s	37ms/step	- loss: 0,0094

Timp (sec) 1411,458660364151

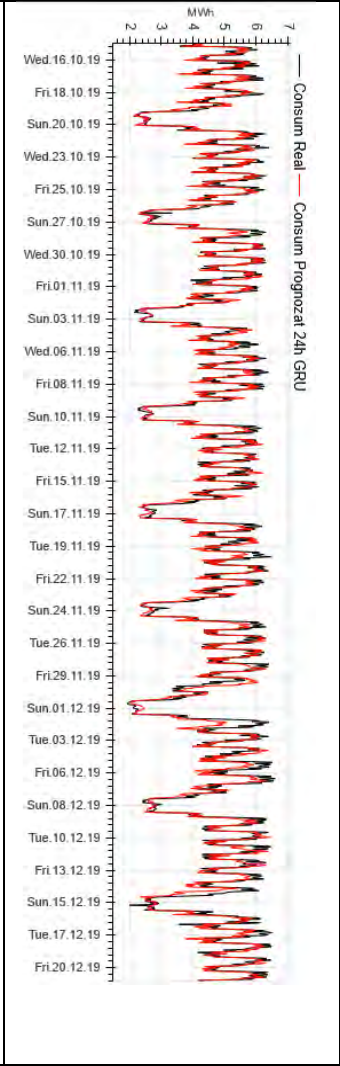
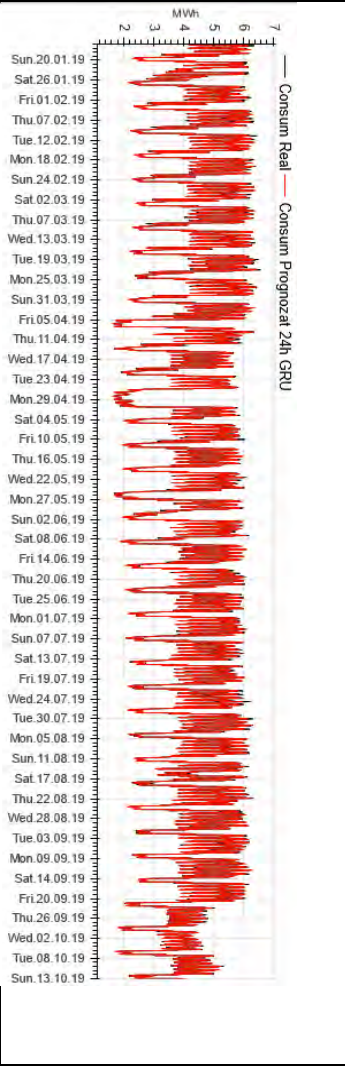
Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0183	-0,016	0,074	-0,004	0,0956	0,0642	-0,134	0,297	-0,026	0,406

```
GRU_LC_agg_24h {GRU[24|5|160|160|60]}
verbose, epochs, batch_size = 1, 200, 24
model_gru = Sequential()
model_gru.add(GRU(160,
return_sequences=True,input_shape=(train_x.shape[1],
train_x.shape[2])))
model_gru.add(GRU(160, return_sequences=True))
model_gru.add(GRU(60, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
```

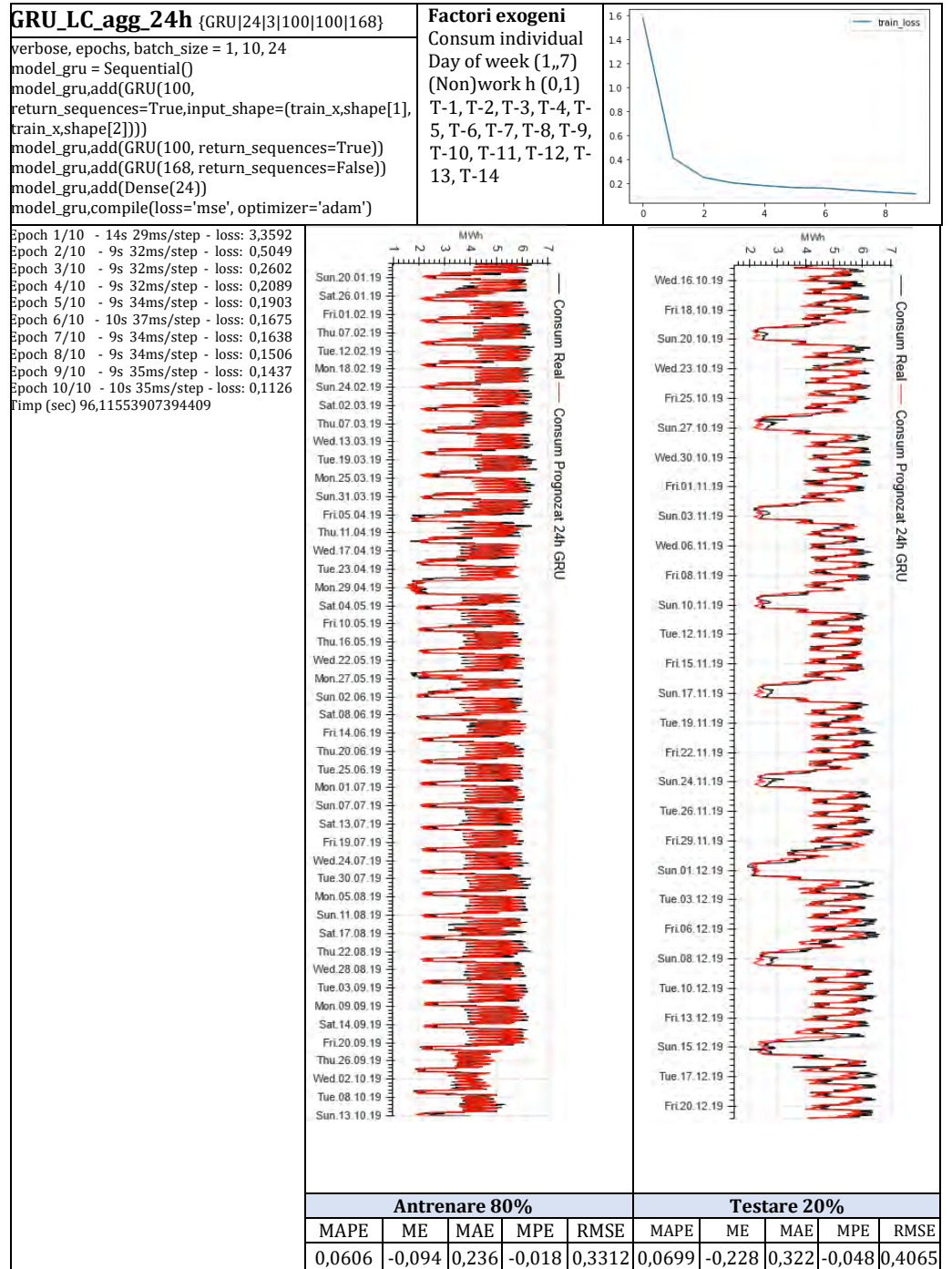
Factori exogeni
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14



Epoch 1/200 - 13s 30ms/step - loss: 4,5404 Epoch 2/200 - 8s 31ms/step - loss: 0,5863 Epoch 3/200 - 8s 31ms/step - loss: 0,3216 Epoch 4/200 - 8s 29ms/step - loss: 0,237
Epoch 5/200 - 9s 32ms/step - loss: 0,2220
Epoch 6/200 - 8s 31ms/step - loss: 0,1822
Epoch 7/200 - 9s 31ms/step - loss: 0,1742
Epoch 8/200 - 9s 33ms/step - loss: 0,1636
Epoch 9/200 - 9s 34ms/step - loss: 0,1613
Epoch 10/200 - 9s 34ms/step - loss: 0,1512
Epoch 11/200 - 8s 31ms/step - loss: 0,1458
Epoch 12/200 - 8s 30ms/step - loss: 0,1219
Epoch 13/200 - 8s 31ms/step - loss: 0,1288
Epoch 14/200 - 9s 33ms/step - loss: 0,1195
Epoch 15/200 - 9s 31ms/step - loss: 0,1131
Epoch 16/200 - 8s 30ms/step - loss: 0,1129
Epoch 17/200 - 9s 32ms/step - loss: 0,0873
Epoch 18/200 - 9s 32ms/step - loss: 0,0924
Epoch 19/200 - 8s 30ms/step - loss: 0,0952
Epoch 20/200 - 9s 33ms/step - loss: 0,0728
Epoch 21/200 - 9s 32ms/step - loss: 0,0784
Epoch 22/200 - 9s 32ms/step - loss: 0,0849
Epoch 23/200 - 9s 32ms/step - loss: 0,0718
Epoch 24/200 - 9s 33ms/step - loss: 0,1026
Epoch 25/200 - 8s 31ms/step - loss: 0,0970
Epoch 26/200 - 9s 33ms/step - loss: 0,0531
Epoch 27/200 - 9s 33ms/step - loss: 0,0542
Epoch 28/200 - 9s 33ms/step - loss: 0,0448
Epoch 29/200 - 9s 32ms/step - loss: 0,0480
Epoch 30/200 - 8s 31ms/step - loss: 0,0461
.....
Epoch 170/200 - 9s 33ms/step - loss: 0,0083
Epoch 171/200 - 8s 31ms/step - loss: 0,0076
Epoch 172/200 - 8s 31ms/step - loss: 0,0071
Epoch 173/200 - 9s 32ms/step - loss: 0,0073
Epoch 174/200 - 9s 32ms/step - loss: 0,0067
Epoch 175/200 - 8s 30ms/step - loss: 0,0068
Epoch 176/200 - 8s 30ms/step - loss: 0,0068
Epoch 177/200 - 8s 31ms/step - loss: 0,0067
Epoch 178/200 - 9s 33ms/step - loss: 0,0068
Epoch 179/200 - 8s 31ms/step - loss: 0,0075
Epoch 180/200 - 9s 34ms/step - loss: 0,0072
Epoch 181/200 - 9s 32ms/step - loss: 0,0071
Epoch 182/200 - 10s 36ms/step - loss: 0,0076
Epoch 183/200 - 8s 30ms/step - loss: 0,0104
Epoch 184/200 - 8s 31ms/step - loss: 0,0085
Epoch 185/200 - 8s 30ms/step - loss: 0,0066
Epoch 186/200 - 8s 31ms/step - loss: 0,0077
Epoch 187/200 - 9s 34ms/step - loss: 0,0073
Epoch 188/200 - 9s 32ms/step - loss: 0,0087
Epoch 189/200 - 8s 31ms/step - loss: 0,0068
Epoch 190/200 - 8s 31ms/step - loss: 0,0070
Epoch 191/200 - 9s 34ms/step - loss: 0,0528
Epoch 192/200 - 9s 33ms/step - loss: 0,0230
Epoch 193/200 - 10s 36ms/step - loss: 0,0131
Epoch 194/200 - 10s 36ms/step - loss: 0,0087
Epoch 195/200 - 10s 36ms/step - loss: 0,0210
Epoch 196/200 - 9s 34ms/step - loss: 0,0140
Epoch 197/200 - 11s 39ms/step - loss: 0,0078
Epoch 198/200 - 10s 35ms/step - loss: 0,0067
Epoch 199/200 - 13s 47ms/step - loss: 0,0066
Epoch 200/200 - 11s 42ms/step - loss: 0,0061
Time (sec) 1752,8610982894897



Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0143	-0,007	0,058	-0,001	0,0746	0,0691	-0,13	0,281	-0,023	0,3806



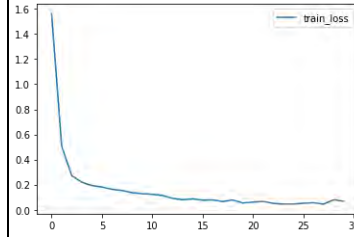
GRU_LC_agg_24h {GRU[24|3|100|100|168]}

```

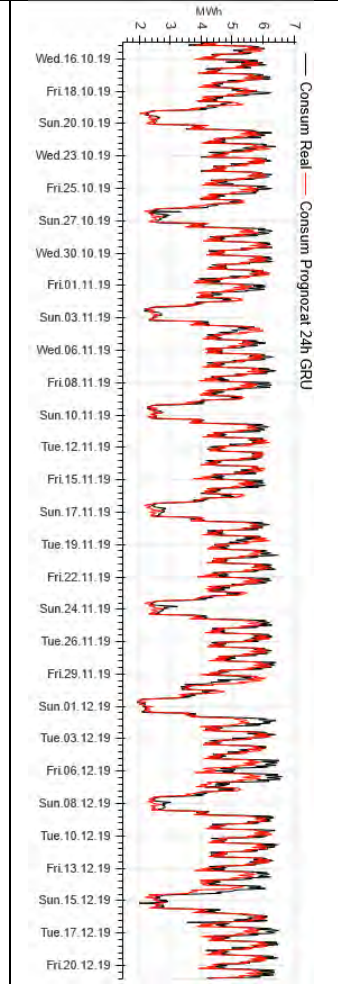
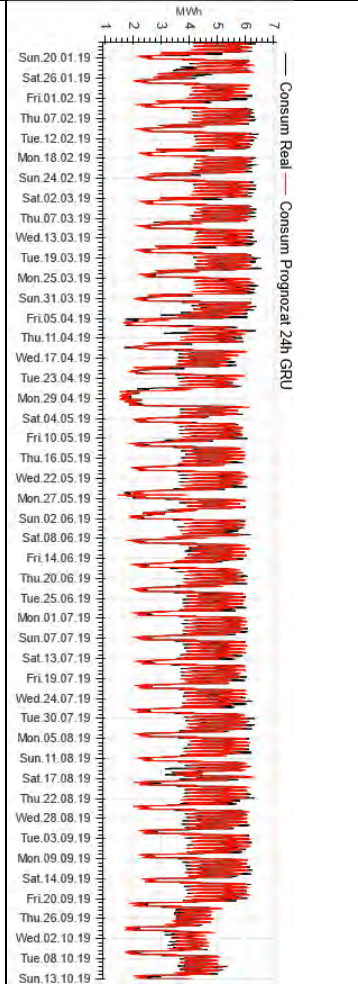
verbose, epochs, batch_size = 1, 30, 24
model_gru = Sequential()
model_gru.add(GRU(100,
return_sequences=True,input_shape=(train_x.shape[1],
train_x.shape[2])))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(168, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
    
```

Factori exogeni

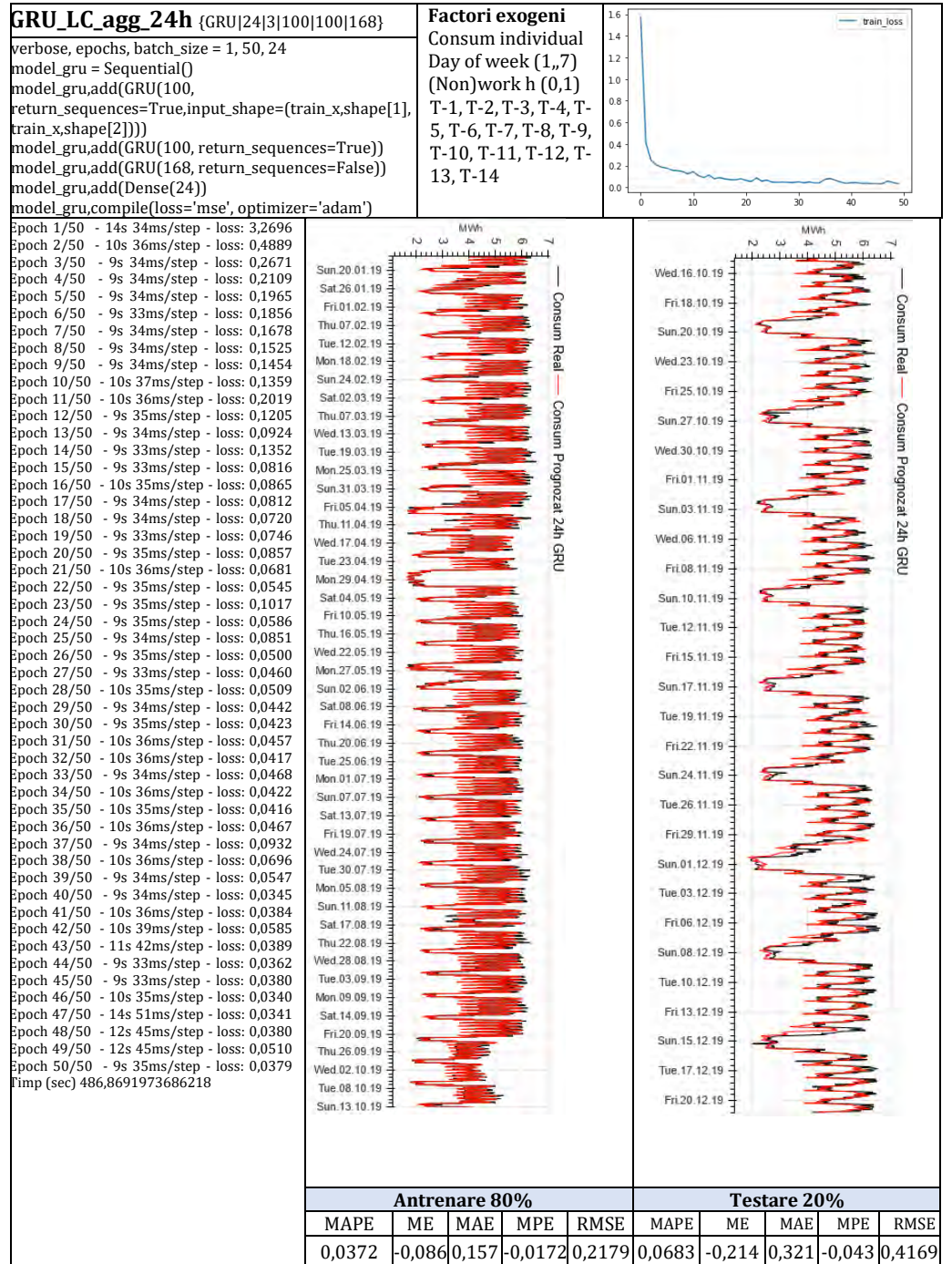
Consum individual
 Day of week (1,,7)
 (Non)work h (0,1)
 T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14



Epoch 1/30 - 13s 29ms/step - loss: 3,2550
 Epoch 2/30 - 9s 32ms/step - loss: 0,5812
 Epoch 3/30 - 9s 33ms/step - loss: 0,2974
 Epoch 4/30 - 9s 34ms/step - loss: 0,2348
 Epoch 5/30 - 9s 32ms/step - loss: 0,1861
 Epoch 6/30 - 9s 32ms/step - loss: 0,1808
 Epoch 7/30 - 9s 33ms/step - loss: 0,1633
 Epoch 8/30 - 10s 36ms/step - loss: 0,1612
 Epoch 9/30 - 9s 33ms/step - loss: 0,1346
 Epoch 10/30 - 10s 37ms/step - loss: 0,1307
 Epoch 11/30 - 8s 30ms/step - loss: 0,1297
 Epoch 12/30 - 8s 29ms/step - loss: 0,1299
 Epoch 13/30 - 8s 31ms/step - loss: 0,0926
 Epoch 14/30 - 8s 31ms/step - loss: 0,0829
 Epoch 15/30 - 8s 31ms/step - loss: 0,1038
 Epoch 16/30 - 9s 32ms/step - loss: 0,0799
 Epoch 17/30 - 9s 32ms/step - loss: 0,0809
 Epoch 18/30 - 8s 31ms/step - loss: 0,0619
 Epoch 19/30 - 9s 31ms/step - loss: 0,0895
 Epoch 20/30 - 9s 33ms/step - loss: 0,0524
 Epoch 21/30 - 8s 31ms/step - loss: 0,0551
 Epoch 22/30 - 9s 32ms/step - loss: 0,0734
 Epoch 23/30 - 8s 30ms/step - loss: 0,0571
 Epoch 24/30 - 8s 30ms/step - loss: 0,0470
 Epoch 25/30 - 8s 30ms/step - loss: 0,0462
 Epoch 26/30 - 8s 30ms/step - loss: 0,0455
 Epoch 27/30 - 8s 31ms/step - loss: 0,0691
 Epoch 28/30 - 8s 31ms/step - loss: 0,0469
 Epoch 29/30 - 8s 31ms/step - loss: 0,0514
 Epoch 30/30 - 9s 34ms/step - loss: 0,0790
 Timp (sec) 263,22988200187683



Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0473	-0,037	0,185	-0,005	0,2627	0,0588	-0,1451	0,277	-0,028	0,357



GRU_LC_agg_24h {GRU[24|3|100|100|168]}

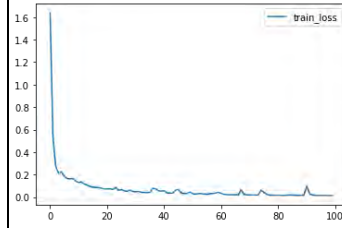
```

verbose, epochs, batch_size = 1, 100, 24
model_gru = Sequential()
model_gru.add(GRU(100,
return_sequences=True,input_shape=(train_x.shape[1],
train_x.shape[2])))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(168, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')

```

Factori exogeni

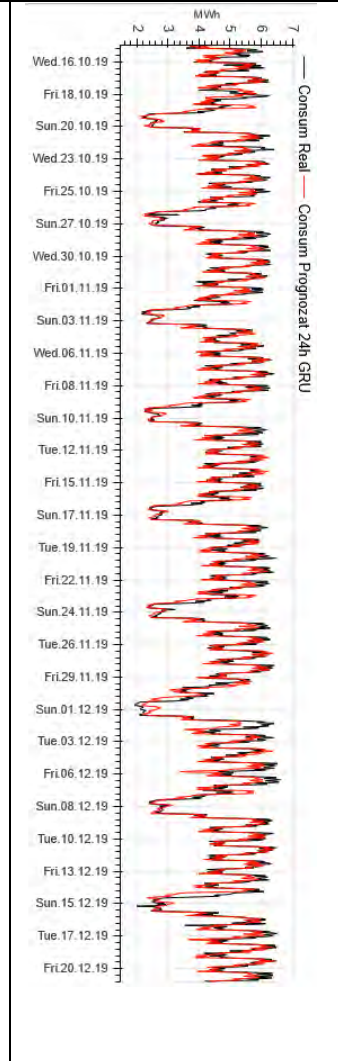
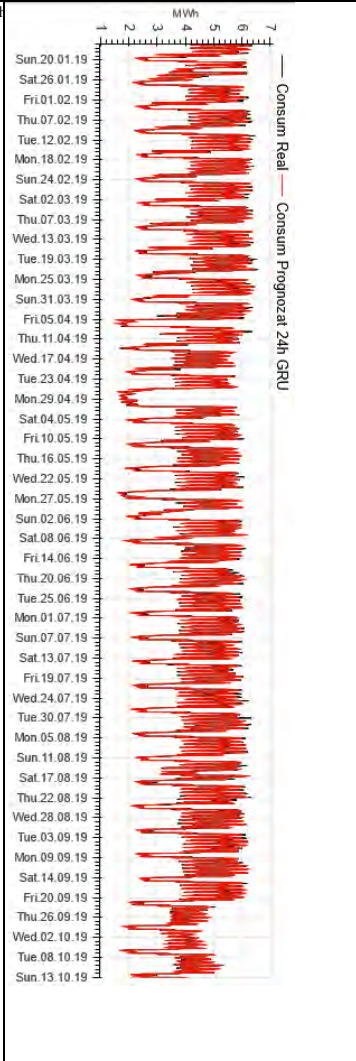
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14



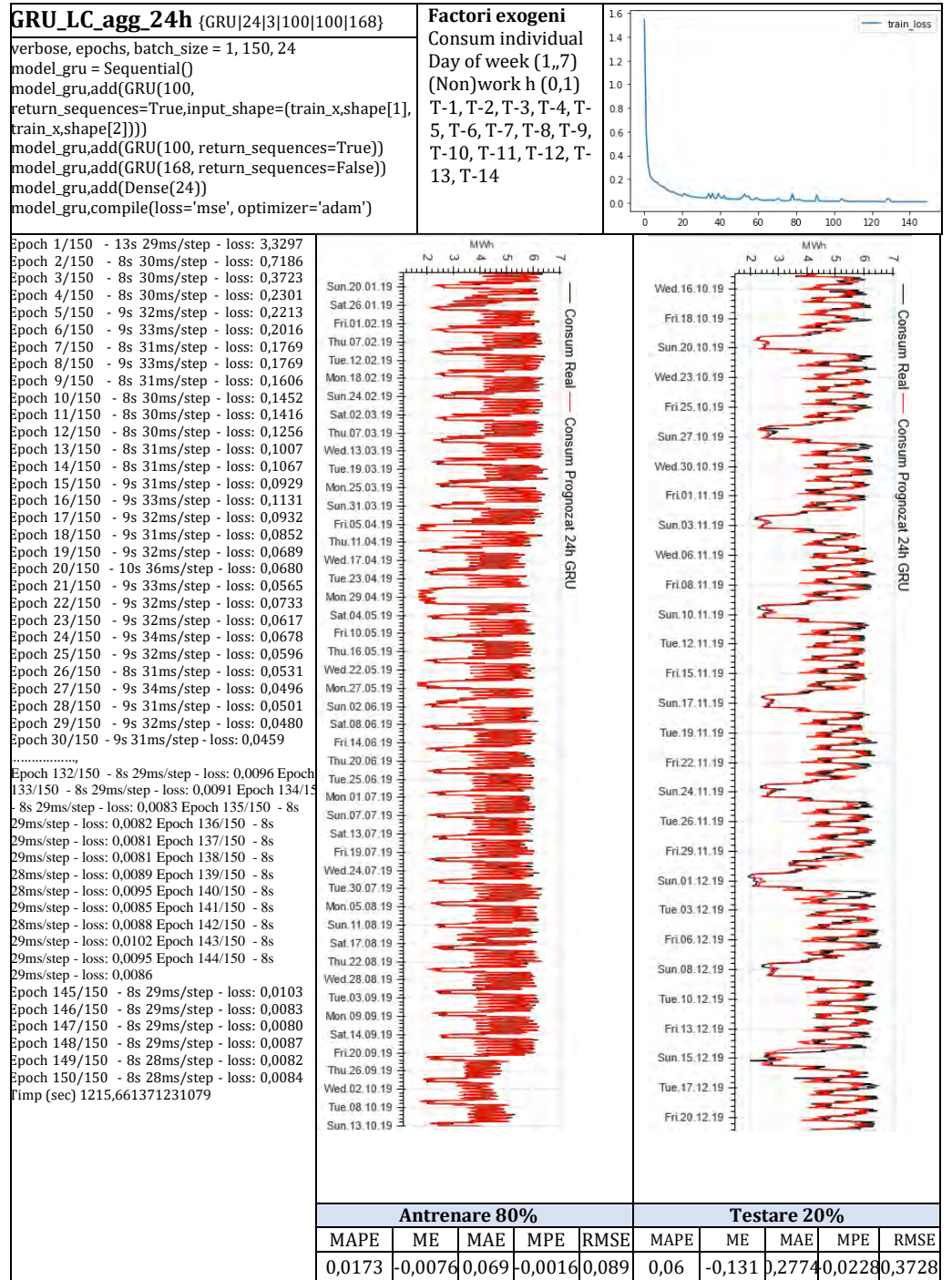
```

Epoch 1/100 - 13s 28ms/step - loss: 3.4327 Epoch
2/100 - 8s 30ms/step - loss: 0.6006 Epoch 3/100
8s 30ms/step - loss: 0.2934 Epoch 4/100 - 8s
30ms/step - loss: 0.2157 Epoch 5/100 - 8s
30ms/step - loss: 0.2988 Epoch 6/100 - 8s
30ms/step - loss: 0.1812 Epoch 7/100 - 9s
32ms/step - loss: 0.1663 Epoch 8/100 - 9s
32ms/step - loss: 0.1661 Epoch 9/100 - 9s
33ms/step - loss: 0.2121 Epoch 10/100 - 9s
34ms/step - loss: 0.1500 Epoch 11/100 - 9s
33ms/step - loss: 0.1470 Epoch 12/100 - 10s
35ms/step - loss: 0.1295 Epoch 13/100 - 9s
34ms/step - loss: 0.1167 Epoch 14/100 - 9s
33ms/step - loss: 0.1233 Epoch 15/100 - 9s
34ms/step - loss: 0.0885 Epoch 16/100 - 10s
37ms/step - loss: 0.0959 Epoch 17/100 - 10s
38ms/step - loss: 0.0867 Epoch 18/100 - 10s
36ms/step - loss: 0.0760 Epoch 19/100 - 10s
36ms/step - loss: 0.0774 Epoch 20/100 - 9s
34ms/step - loss: 0.0734 Epoch 21/100 - 8s
31ms/step - loss: 0.0686 Epoch 22/100 - 8s
31ms/step - loss: 0.0739 Epoch 23/100 - 8s
31ms/step - loss: 0.0627 Epoch 24/100 - 8s
31ms/step - loss: 0.0934 Epoch 25/100 - 8s
31ms/step - loss: 0.0635 Epoch 26/100 - 9s
32ms/step - loss: 0.0643 Epoch 27/100 - 9s
33ms/step - loss: 0.0643 Epoch 28/100 - 9s
32ms/step - loss: 0.0559 Epoch 29/100 - 9s
33ms/step - loss: 0.0588 Epoch 30/100 - 9s
32ms/step - loss: 0.0566 Epoch 31/100 - 9s
32ms/step - loss: 0.0612 Epoch 32/100 - 9s
34ms/step - loss: 0.0536 Epoch 33/100 - 9s
32ms/step - loss: 0.0420 Epoch 34/100 - 9s
33ms/step - loss: 0.0424 Epoch 35/100 - 9s
32ms/step - loss: 0.0426 Epoch 36/100 - 9s
33ms/step - loss: 0.0399 Epoch 37/100 - 9s
34ms/step - loss: 0.0622 Epoch 38/100 - 9s
32ms/step - loss: 0.0714 Epoch 39/100 - 9s
33ms/step - loss: 0.0607 Epoch 40/100 - 9s
32ms/step - loss: 0.0494
.....
Epoch 79/100 - 9s 31ms/step - loss: 0,0178
Epoch 80/100 - 9s 32ms/step - loss: 0,0160
Epoch 81/100 - 9s 33ms/step - loss: 0,0160
Epoch 82/100 - 8s 31ms/step - loss: 0,0162
Epoch 83/100 - 8s 29ms/step - loss: 0,0165
Epoch 84/100 - 8s 31ms/step - loss: 0,0170
Epoch 85/100 - 8s 29ms/step - loss: 0,0218
Epoch 86/100 - 8s 29ms/step - loss: 0,0179
Epoch 87/100 - 8s 30ms/step - loss: 0,0180
Epoch 88/100 - 9s 32ms/step - loss: 0,0157
Epoch 89/100 - 8s 30ms/step - loss: 0,0155
Epoch 90/100 - 8s 30ms/step - loss: 0,0156
Epoch 91/100 - 8s 30ms/step - loss: 0,0129
Epoch 92/100 - 9s 31ms/step - loss: 0,0393
Epoch 93/100 - 8s 30ms/step - loss: 0,0194
Epoch 94/100 - 8s 30ms/step - loss: 0,0167
Epoch 95/100 - 8s 30ms/step - loss: 0,0151
Epoch 96/100 - 9s 31ms/step - loss: 0,0147
Epoch 97/100 - 8s 29ms/step - loss: 0,0166
Epoch 98/100 - 9s 32ms/step - loss: 0,0144
Epoch 99/100 - 8s 31ms/step - loss: 0,0142
Epoch 100/100 - 9s 32ms/step - loss: 0,0138
Time(sec) 872,3507099151611

```



Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0254	-0,036	0,102	-0,0097	0,1307	0,065	-0,152	0,304	-0,028	0,4002



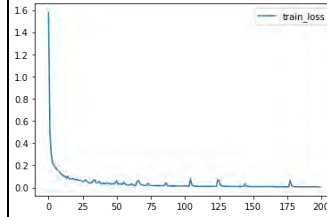
GRU_LC_agg_24h {GRU[24|3|100|100|168]}

```

verbose, epochs, batch_size = 1, 200, 24
model_gru = Sequential()
model_gru.add(GRU(100,
return_sequences=True,input_shape=(train_x.shape[1],
train_x.shape[2])))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(168, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
    
```

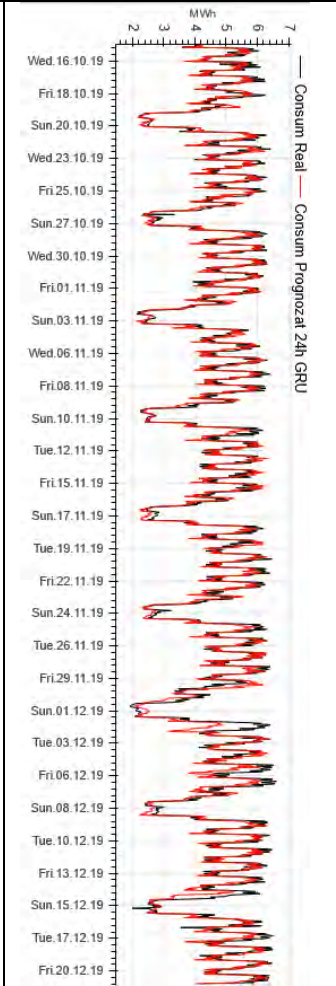
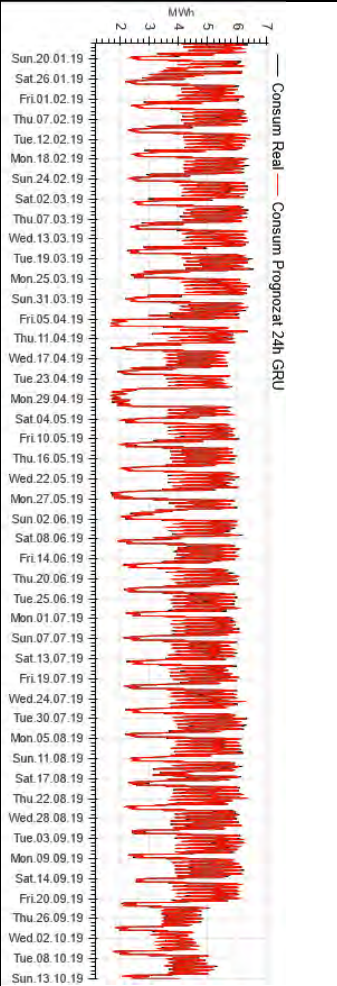
Factori exogeni

Consum individual
 Day of week (1,,7)
 (Non)work h (0,1)
 T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14

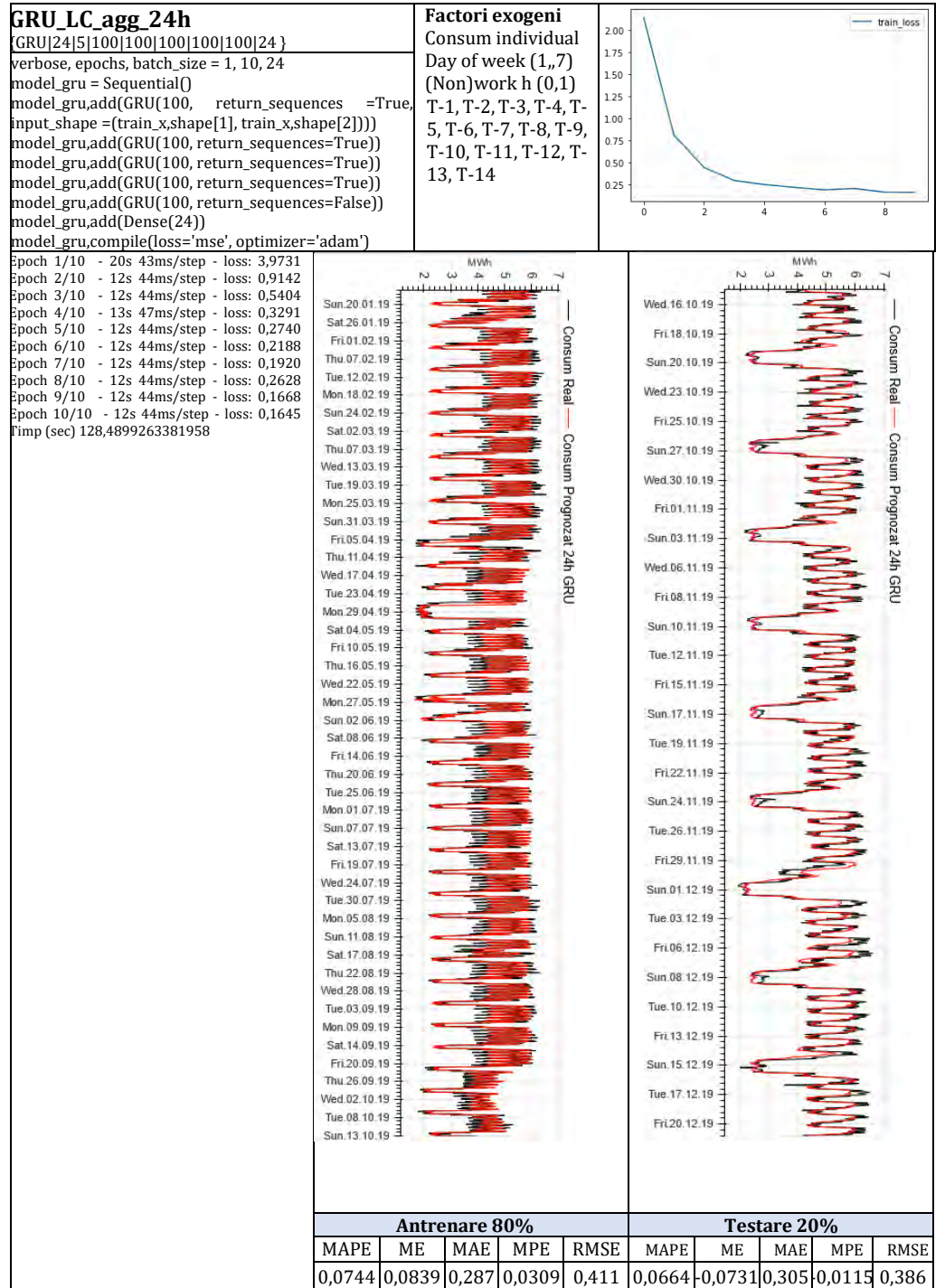


Epoch 1/200 - 14s 31ms/step - loss: 3,3777
 Epoch 2/200 - 7s 27ms/step - loss: 0,5325
 Epoch 3/200 - 7s 27ms/step - loss: 0,3930
 Epoch 4/200 - 7s 27ms/step - loss: 0,2356
 Epoch 5/200 - 7s 28ms/step - loss: 0,2051
 Epoch 6/200 - 9s 32ms/step - loss: 0,1897
 Epoch 7/200 - 11s 41ms/step - loss: 0,1747
 Epoch 8/200 - 9s 34ms/step - loss: 0,1583
 Epoch 9/200 - 9s 31ms/step - loss: 0,1524
 Epoch 10/200 - 8s 31ms/step - loss: 0,1376
 Epoch 11/200 - 9s 31ms/step - loss: 0,1154
 Epoch 12/200 - 9s 32ms/step - loss: 0,1048
 Epoch 13/200 - 9s 32ms/step - loss: 0,1111
 Epoch 14/200 - 9s 34ms/step - loss: 0,0813
 Epoch 15/200 - 9s 33ms/step - loss: 0,1182
 Epoch 16/200 - 9s 32ms/step - loss: 0,0821
 Epoch 17/200 - 9s 33ms/step - loss: 0,0723
 Epoch 18/200 - 9s 33ms/step - loss: 0,0929
 Epoch 19/200 - 9s 32ms/step - loss: 0,0825
 Epoch 20/200 - 9s 32ms/step - loss: 0,0727
 Epoch 21/200 - 9s 32ms/step - loss: 0,0587
 Epoch 22/200 - 9s 32ms/step - loss: 0,0816
 Epoch 23/200 - 9s 32ms/step - loss: 0,0776
 Epoch 24/200 - 9s 33ms/step - loss: 0,0609
 Epoch 25/200 - 9s 32ms/step - loss: 0,0720
 Epoch 26/200 - 9s 32ms/step - loss: 0,0502
 Epoch 27/200 - 9s 32ms/step - loss: 0,0558:
 Epoch 28/200 - 9s 32ms/step - loss: 0,0738
 Epoch 29/200 - 9s 32ms/step - loss: 0,0583
 Epoch 30/200 - 9s 35ms/step - loss: 0,0477

 Epoch 180/200 - 9s 33ms/step - loss: 0,0115
 Epoch 181/200 - 9s 34ms/step - loss: 0,0078
 Epoch 182/200 - 9s 34ms/step - loss: 0,0076
 Epoch 183/200 - 10s 36ms/step - loss: 0,0072
 Epoch 184/200 - 10s 36ms/step - loss: 0,0068
 Epoch 185/200 - 10s 35ms/step - loss: 0,0060
 Epoch 186/200 - 10s 37ms/step - loss: 0,0065
 Epoch 187/200 - 10s 37ms/step - loss: 0,0065
 Epoch 188/200 - 9s 32ms/step - loss: 0,0058
 Epoch 189/200 - 9s 32ms/step - loss: 0,0060
 Epoch 190/200 - 8s 30ms/step - loss: 0,0062
 Epoch 191/200 - 9s 32ms/step - loss: 0,0064
 Epoch 192/200 - 8s 29ms/step - loss: 0,0065
 Epoch 193/200 - 8s 31ms/step - loss: 0,0061
 Epoch 194/200 - 8s 31ms/step - loss: 0,0064
 Epoch 195/200 - 8s 31ms/step - loss: 0,0073
 Epoch 196/200 - 9s 33ms/step - loss: 0,0074
 Epoch 197/200 - 9s 31ms/step - loss: 0,0062
 Epoch 198/200 - 8s 31ms/step - loss: 0,0099
 Epoch 199/200 - 8s 30ms/step - loss: 0,0059
 Epoch 200/200 - 9s 32ms/step - loss: 0,0061
 Timp (sec) 1870,8138904571533



Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0152	-0,0012	0,063	0,0006	0,0818	0,0612	-0,142	0,2894	0,0268	0,3994



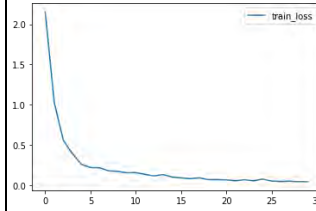
GRU_LC_agg_24h

```
{GRU[24][5][100][100][100][100][24]}
```

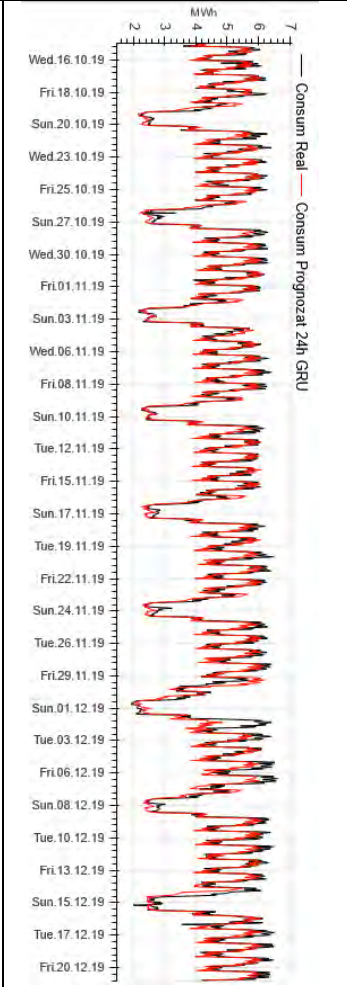
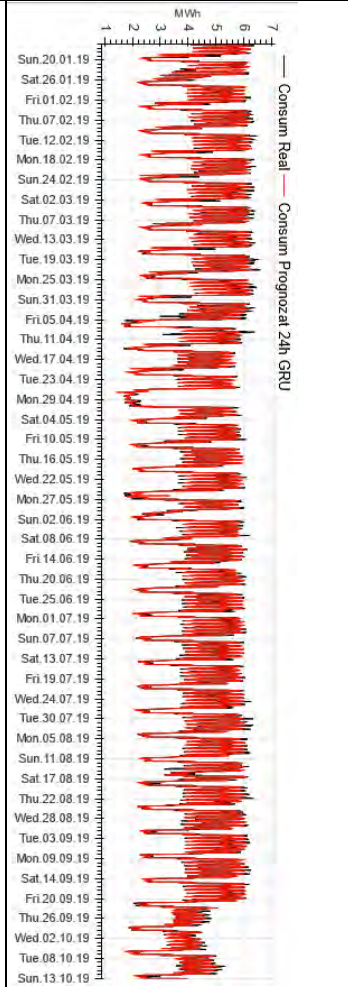
```
verbose, epochs, batch_size = 1, 30, 24
model_gru = Sequential()
model_gru.add(GRU(100, return_sequences =True,
input_shape =(train_x.shape[1], train_x.shape[2])))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(100, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
```

Factori exogeni

Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14



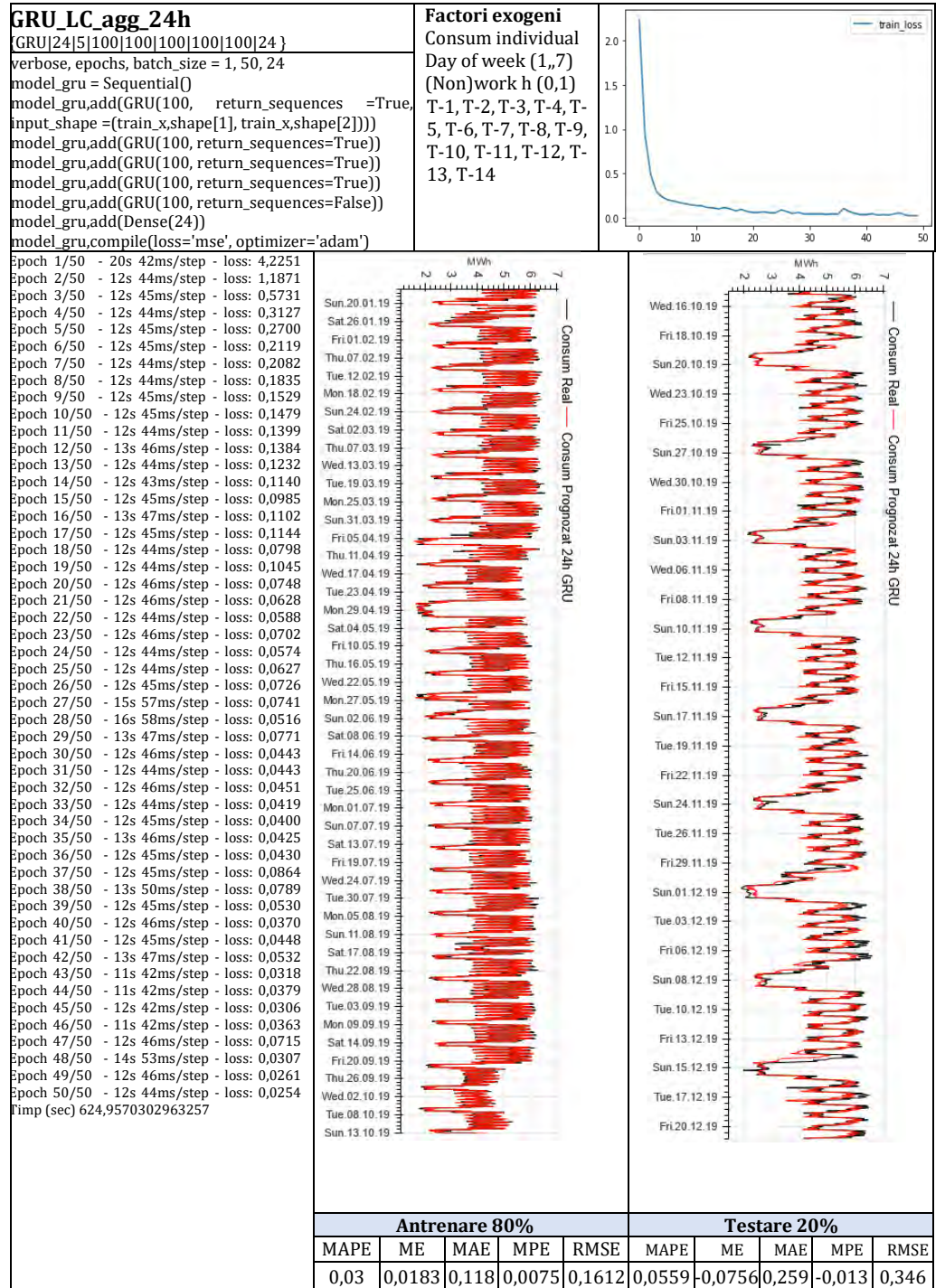
```
Epoch 1/30 - 19s 40ms/step - loss: 3,9685
Epoch 2/30 - 12s 43ms/step - loss: 1,2509
Epoch 3/30 - 12s 44ms/step - loss: 0,5841
Epoch 4/30 - 12s 45ms/step - loss: 0,4475
Epoch 5/30 - 12s 44ms/step - loss: 0,2976
Epoch 6/30 - 12s 43ms/step - loss: 0,2187
Epoch 7/30 - 11s 42ms/step - loss: 0,2461
Epoch 8/30 - 12s 43ms/step - loss: 0,1912
Epoch 9/30 - 12s 45ms/step - loss: 0,1820
Epoch 10/30 - 12s 45ms/step - loss: 0,1659
Epoch 11/30 - 12s 44ms/step - loss: 0,1537
Epoch 12/30 - 13s 48ms/step - loss: 0,1436
Epoch 13/30 - 13s 46ms/step - loss: 0,1180
Epoch 14/30 - 12s 45ms/step - loss: 0,1395
Epoch 15/30 - 12s 46ms/step - loss: 0,1058
Epoch 16/30 - 12s 43ms/step - loss: 0,0978
Epoch 17/30 - 12s 45ms/step - loss: 0,0839
Epoch 18/30 - 13s 46ms/step - loss: 0,0875
Epoch 19/30 - 12s 45ms/step - loss: 0,0653
Epoch 20/30 - 13s 46ms/step - loss: 0,0690
Epoch 21/30 - 12s 45ms/step - loss: 0,0613
Epoch 22/30 - 12s 46ms/step - loss: 0,0557
Epoch 23/30 - 13s 46ms/step - loss: 0,0538
Epoch 24/30 - 13s 47ms/step - loss: 0,0595
Epoch 25/30 - 13s 48ms/step - loss: 0,0622
Epoch 26/30 - 13s 48ms/step - loss: 0,0524
Epoch 27/30 - 13s 47ms/step - loss: 0,0458
Epoch 28/30 - 13s 46ms/step - loss: 0,0581
Epoch 29/30 - 13s 49ms/step - loss: 0,0443
Epoch 30/30 - 12s 43ms/step - loss: 0,0467
Time (sec) 376,13020491600037
```



Antrenare 80%

Testare 20%

MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0369	-0,0519	0,149	-0,01	0,213	0,0664	-0,073	0,305	-0,0115	0,386

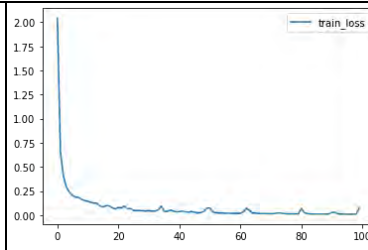


GRU_LC_agg_24h

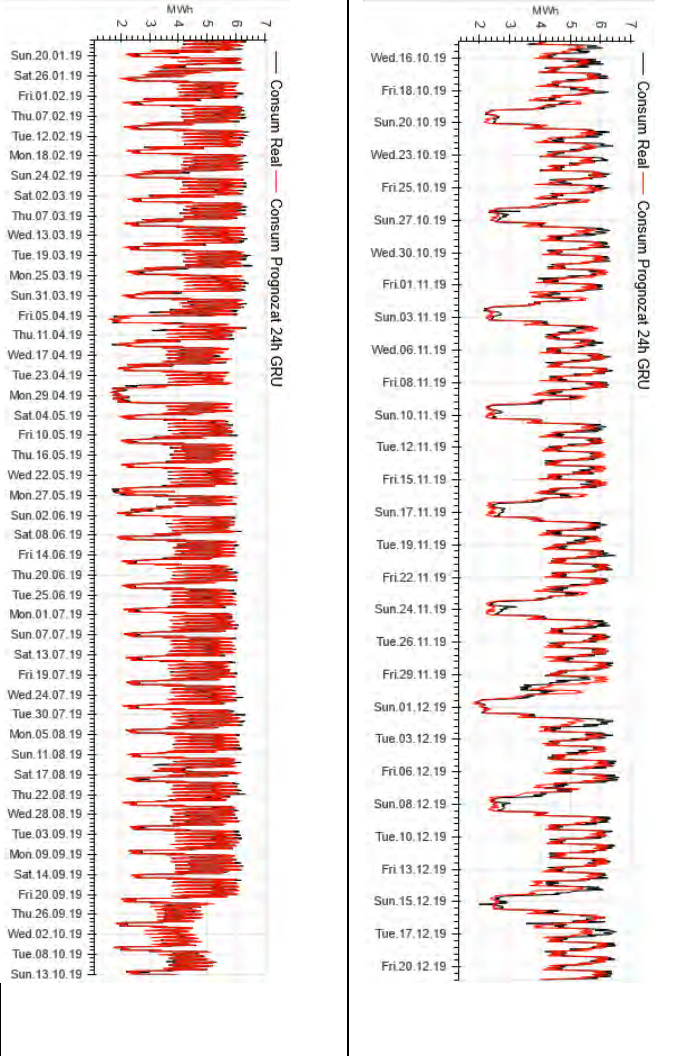
```
GRU[24|5|100|100|100|100|24]
verbose, epochs, batch_size = 1, 100, 24
model_gru = Sequential()
model_gru.add(GRU(100, return_sequences =True,
input_shape =(train_x.shape[1], train_x.shape[2])))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(100, return_sequences=True))
model_gru.add(GRU(100, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
```

Factori exogeni

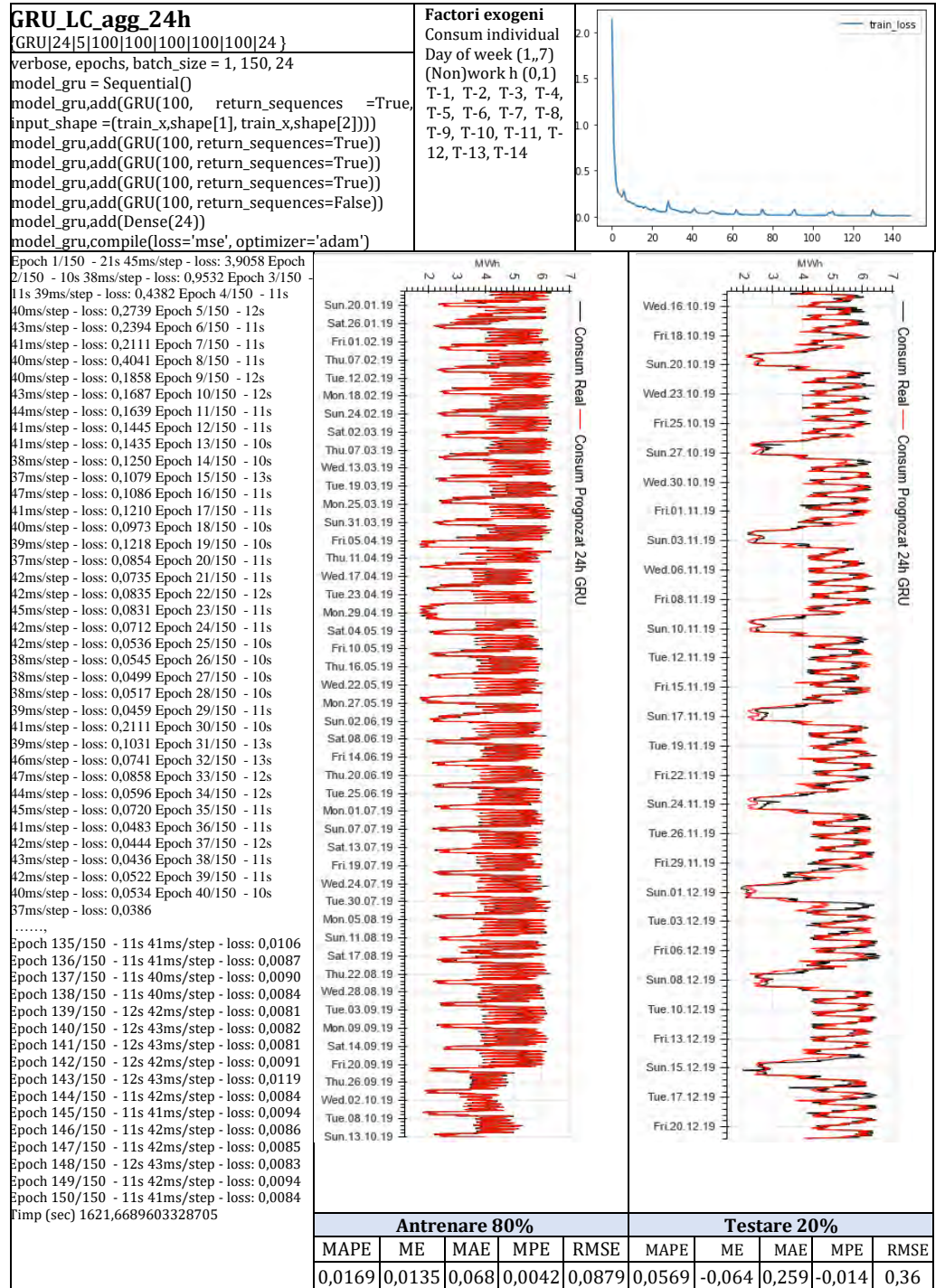
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14

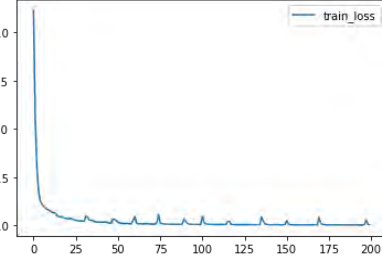
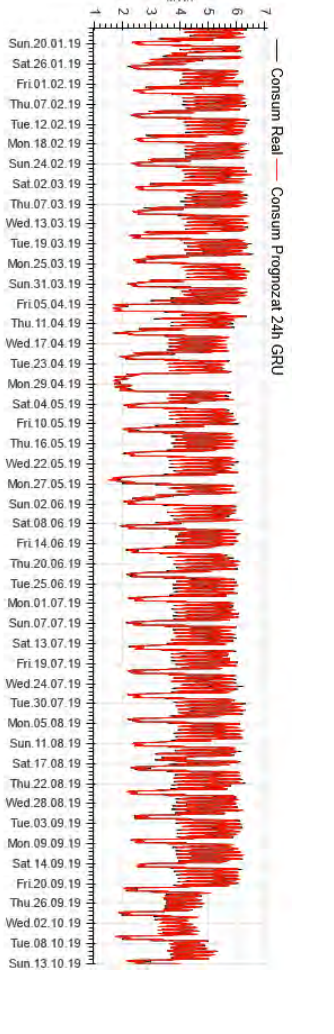
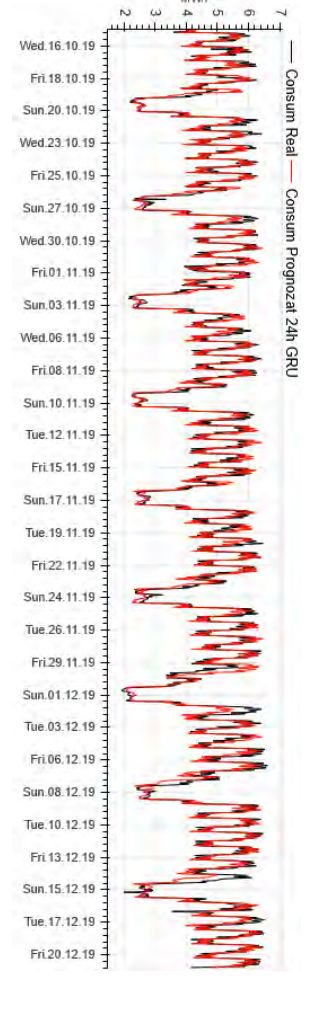


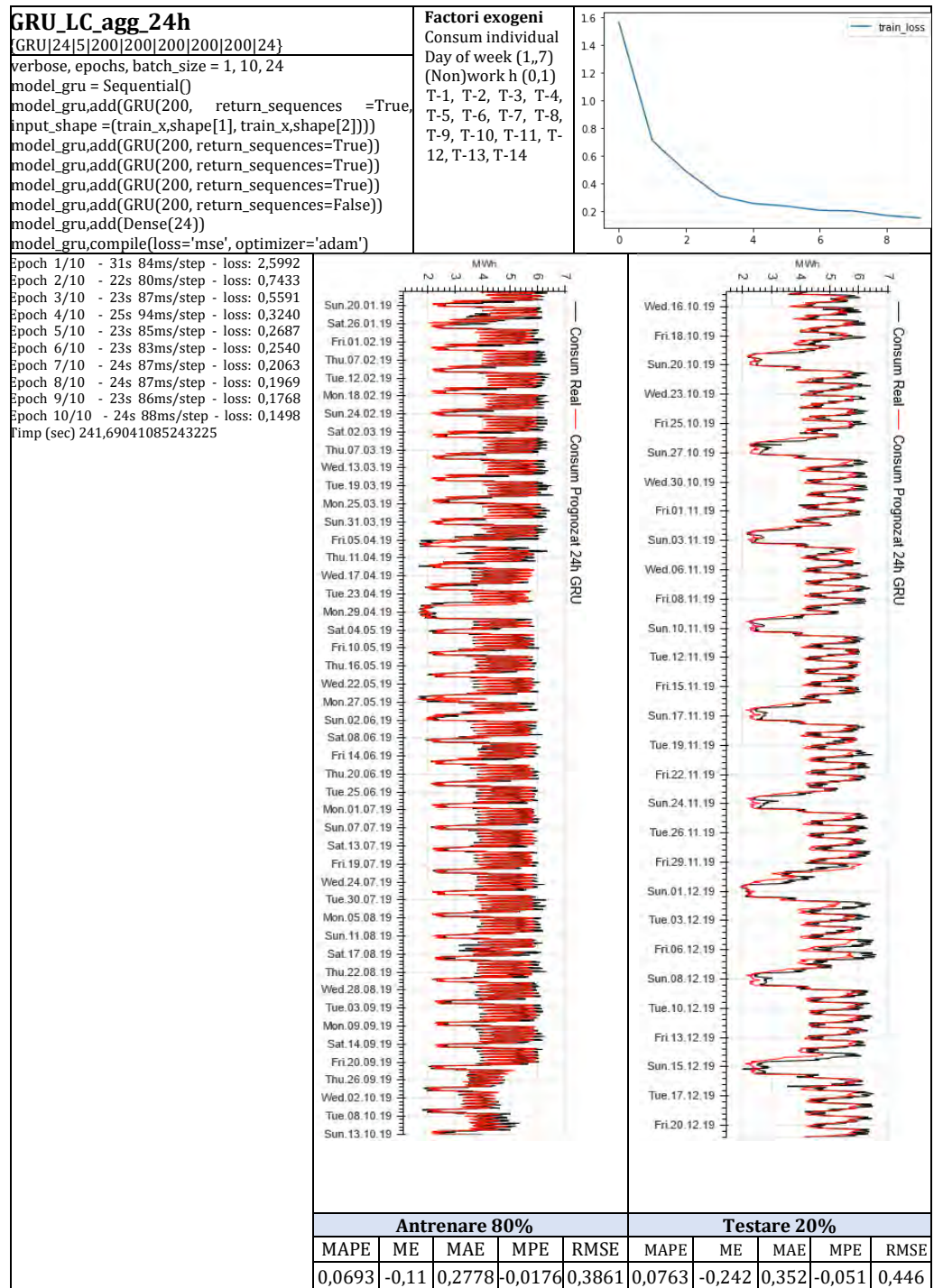
Epoch 1/100 - 21s 44ms/step - loss: 4.0904 Epoch 2/100 - 13s 49ms/step - loss: 0.7306 Epoch 3/100 - 14s 50ms/step - loss: 0.4840 Epoch 4/100 - 13s 47ms/step - loss: 0.3256 Epoch 5/100 - 13s 49ms/step - loss: 0.2520 Epoch 6/100 - 14s 52ms/step - loss: 0.2035 Epoch 7/100 - 13s 48ms/step - loss: 0.1981 Epoch 8/100 - 14s 52ms/step - loss: 0.1791 Epoch 9/100 - 13s 49ms/step - loss: 0.1683 Epoch 10/100 - 14s 50ms/step - loss: 0.1543 Epoch 11/100 - 13s 49ms/step - loss: 0.1514 Epoch 12/100 - 14s 51ms/step - loss: 0.1391 Epoch 13/100 - 14s 53ms/step - loss: 0.1225 Epoch 14/100 - 14s 50ms/step - loss: 0.1260 Epoch 15/100 - 12s 44ms/step - loss: 0.1016 Epoch 16/100 - 12s 44ms/step - loss: 0.0865 Epoch 17/100 - 12s 44ms/step - loss: 0.0855 Epoch 18/100 - 12s 44ms/step - loss: 0.1180 Epoch 19/100 - 12s 44ms/step - loss: 0.0873 Epoch 20/100 - 14s 51ms/step - loss: 0.0734 Epoch 21/100 - 12s 46ms/step - loss: 0.0696 Epoch 22/100 - 13s 47ms/step - loss: 0.0648 Epoch 23/100 - 12s 46ms/step - loss: 0.0934 Epoch 24/100 - 13s 47ms/step - loss: 0.0853 Epoch 25/100 - 14s 51ms/step - loss: 0.0700 Epoch 26/100 - 13s 46ms/step - loss: 0.0494 Epoch 27/100 - 13s 47ms/step - loss: 0.0509 Epoch 28/100 - 12s 44ms/step - loss: 0.0478 Epoch 29/100 - 12s 45ms/step - loss: 0.0462 Epoch 30/100 - 12s 44ms/step - loss: 0.0425 Epoch 31/100 - 13s 46ms/step - loss: 0.0455 Epoch 32/100 - 13s 47ms/step - loss: 0.0437 Epoch 33/100 - 14s 51ms/step - loss: 0.0434 Epoch 34/100 - 13s 47ms/step - loss: 0.0463 Epoch 35/100 - 13s 46ms/step - loss: 0.1074 Epoch 36/100 - 12s 46ms/step - loss: 0.0444
.....
Epoch 76/100 - 12s 43ms/step - loss: 0,0150
Epoch 77/100 - 12s 43ms/step - loss: 0,0146
Epoch 78/100 - 12s 43ms/step - loss: 0,0137
Epoch 79/100 - 12s 43ms/step - loss: 0,0146
Epoch 80/100 - 12s 43ms/step - loss: 0,0137
Epoch 81/100 - 12s 43ms/step - loss: 0,0754
Epoch 82/100 - 12s 44ms/step - loss: 0,0304
Epoch 83/100 - 12s 43ms/step - loss: 0,0167
Epoch 84/100 - 12s 43ms/step - loss: 0,0140
Epoch 85/100 - 12s 43ms/step - loss: 0,0134
Epoch 86/100 - 12s 43ms/step - loss: 0,0122
Epoch 87/100 - 12s 43ms/step - loss: 0,0120
Epoch 88/100 - 12s 43ms/step - loss: 0,0116
Epoch 89/100 - 12s 43ms/step - loss: 0,0125
Epoch 90/100 - 12s 43ms/step - loss: 0,0122
Epoch 91/100 - 12s 43ms/step - loss: 0,0146
Epoch 92/100 - 12s 44ms/step - loss: 0,0479
Epoch 93/100 - 12s 43ms/step - loss: 0,0137
Epoch 94/100 - 12s 43ms/step - loss: 0,0116
Epoch 95/100 - 12s 44ms/step - loss: 0,0110
Epoch 96/100 - 12s 43ms/step - loss: 0,0109
Epoch 97/100 - 12s 43ms/step - loss: 0,0105
Epoch 98/100 - 12s 45ms/step - loss: 0,0113
Epoch 99/100 - 12s 45ms/step - loss: 0,0110
Epoch 100/100 - 12s 45ms/step - loss: 0,0913
Timp (sec) 1245,315759420395



Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0313	-0,02	0,121	-0,004	0,1697	0,0618	-0,1156	0,285	-0,025	0,3685



GRU_LC_agg_24h {GRU[24][5][100][100][100][100][24]} verbose, epochs, batch_size = 1, 200, 24 model_gru = Sequential() model_gru.add(GRU(100, return_sequences =True, input_shape=(train_x.shape[1], train_x.shape[2]))) model_gru.add(GRU(100, return_sequences=True)) model_gru.add(GRU(100, return_sequences=True)) model_gru.add(GRU(100, return_sequences=True)) model_gru.add(GRU(100, return_sequences=False)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam')	Factori exogeni Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T-12, T-13, T-14																															
Epoch 1/200 - 18s 34ms/step - loss: 4,1806 Epoch 2/200 - 11s 39ms/step - loss: 1,4121 Epoch 3/200 - 11s 39ms/step - loss: 0,6665 Epoch 4/200 - 10s 38ms/step - loss: 0,4243 Epoch 5/200 - 13s 46ms/step - loss: 0,2681 Epoch 6/200 - 11s 39ms/step - loss: 0,2345 Epoch 7/200 - 10s 37ms/step - loss: 0,2262 Epoch 8/200 - 10s 38ms/step - loss: 0,1934 Epoch 9/200 - 11s 39ms/step - loss: 0,1711 Epoch 10/200 - 11s 42ms/step - loss: 0,1590 Epoch 11/200 - 11s 42ms/step - loss: 0,1691 Epoch 12/200 - 11s 39ms/step - loss: 0,1516 Epoch 13/200 - 11s 39ms/step - loss: 0,1266 Epoch 14/200 - 11s 39ms/step - loss: 0,1258 Epoch 15/200 - 10s 38ms/step - loss: 0,1061 Epoch 16/200 - 10s 39ms/step - loss: 0,0972 Epoch 17/200 - 11s 39ms/step - loss: 0,0855 Epoch 18/200 - 10s 39ms/step - loss: 0,1095 Epoch 19/200 - 11s 39ms/step - loss: 0,0807 Epoch 20/200 - 11s 41ms/step - loss: 0,0789 Epoch 21/200 - 11s 39ms/step - loss: 0,0844 Epoch 22/200 - 11s 41ms/step - loss: 0,0636 Epoch 23/200 - 12s 44ms/step - loss: 0,0891 Epoch 24/200 - 11s 40ms/step - loss: 0,0633 Epoch 25/200 - 12s 43ms/step - loss: 0,0612 Epoch 26/200 - 11s 41ms/step - loss: 0,0535 Epoch 27/200 - 11s 39ms/step - loss: 0,0505 Epoch 28/200 - 11s 42ms/step - loss: 0,0488 Epoch 29/200 - 11s 39ms/step - loss: 0,0481 Epoch 30/200 - 11s 40ms/step - loss: 0,0509 Epoch 31/200 - 13s 47ms/step - loss: 0,0443 Epoch 32/200 - 12s 43ms/step - loss: 0,0633 Epoch 33/200 - 12s 43ms/step - loss: 0,1279 Epoch 34/200 - 12s 44ms/step - loss: 0,0536 Epoch 35/200 - 12s 45ms/step - loss: 0,0455 Epoch 36/200 - 11s 40ms/step - loss: 0,0654 Epoch 37/200 - 11s 40ms/step - loss: 0,0408 Epoch 38/200 - 11s 41ms/step - loss: 0,0404 Epoch 39/200 - 12s 46ms/step - loss: 0,0382 Epoch 40/200 - 11s 40ms/step - loss: 0,0349																																
Epoch 180/200 - 11s 39ms/step - loss: 0,0072 Epoch 181/200 - 10s 36ms/step - loss: 0,0063 Epoch 182/200 - 11s 40ms/step - loss: 0,0062 Epoch 183/200 - 10s 38ms/step - loss: 0,0068 Epoch 184/200 - 11s 41ms/step - loss: 0,0063 Epoch 185/200 - 10s 38ms/step - loss: 0,0062 Epoch 186/200 - 10s 38ms/step - loss: 0,0077 Epoch 187/200 - 10s 38ms/step - loss: 0,0076 Epoch 188/200 - 11s 39ms/step - loss: 0,0064 Epoch 189/200 - 10s 38ms/step - loss: 0,0068 Epoch 190/200 - 10s 38ms/step - loss: 0,0067 Epoch 191/200 - 10s 37ms/step - loss: 0,0073 Epoch 192/200 - 10s 38ms/step - loss: 0,0067 Epoch 193/200 - 11s 39ms/step - loss: 0,0069 Epoch 194/200 - 10s 38ms/step - loss: 0,0076 Epoch 195/200 - 11s 40ms/step - loss: 0,0060 Epoch 196/200 - 10s 38ms/step - loss: 0,0056 Epoch 197/200 - 10s 38ms/step - loss: 0,0061 Epoch 198/200 - 10s 37ms/step - loss: 0,0087 Epoch 199/200 - 11s 41ms/step - loss: 0,0154 Epoch 200/200 - 10s 38ms/step - loss: 0,0083 timp (sec) 2196,4808814525604	<table border="1"> <thead> <tr> <th colspan="5">Antrenare 80%</th> <th colspan="5">Testare 20%</th> </tr> <tr> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> </tr> </thead> <tbody> <tr> <td>0,0156</td> <td>0,012</td> <td>0,0634</td> <td>0,0026</td> <td>0,0814</td> <td>0,0569</td> <td>-0,064</td> <td>0,259</td> <td>-0,014</td> <td>0,36</td> </tr> </tbody> </table>		Antrenare 80%					Testare 20%					MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE	0,0156	0,012	0,0634	0,0026	0,0814	0,0569	-0,064	0,259	-0,014	0,36
Antrenare 80%					Testare 20%																											
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE																							
0,0156	0,012	0,0634	0,0026	0,0814	0,0569	-0,064	0,259	-0,014	0,36																							

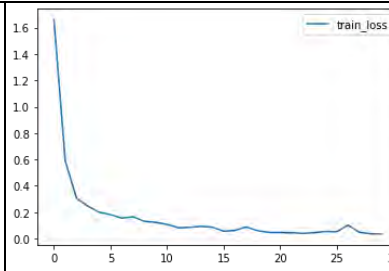


GRU_LC_agg_24h

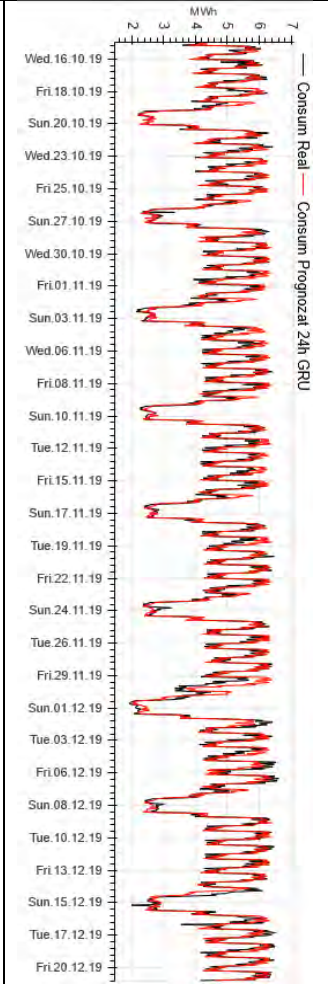
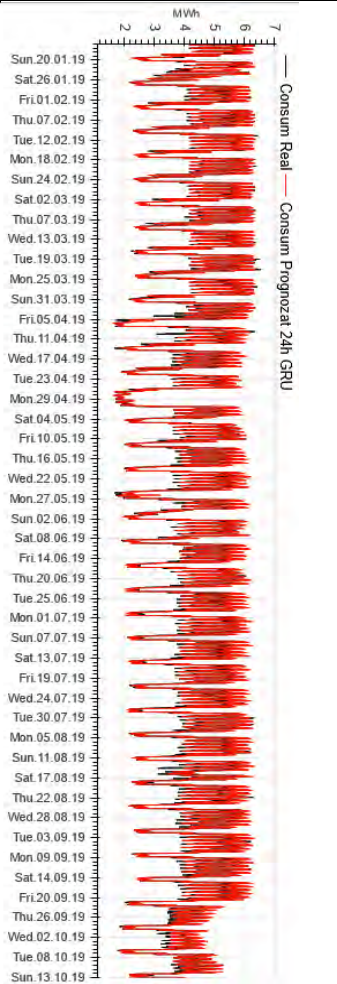
```
{GRU[24|5|200|200|200|200|24]}
verbose, epochs, batch_size = 1, 30, 24
model_gru = Sequential()
model_gru.add(GRU(200, return_sequences =True,
input_shape =(train_x.shape[1], train_x.shape[2])))
model_gru.add(GRU(200, return_sequences=True))
model_gru.add(GRU(200, return_sequences=True))
model_gru.add(GRU(200, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
```

Factori exogeni

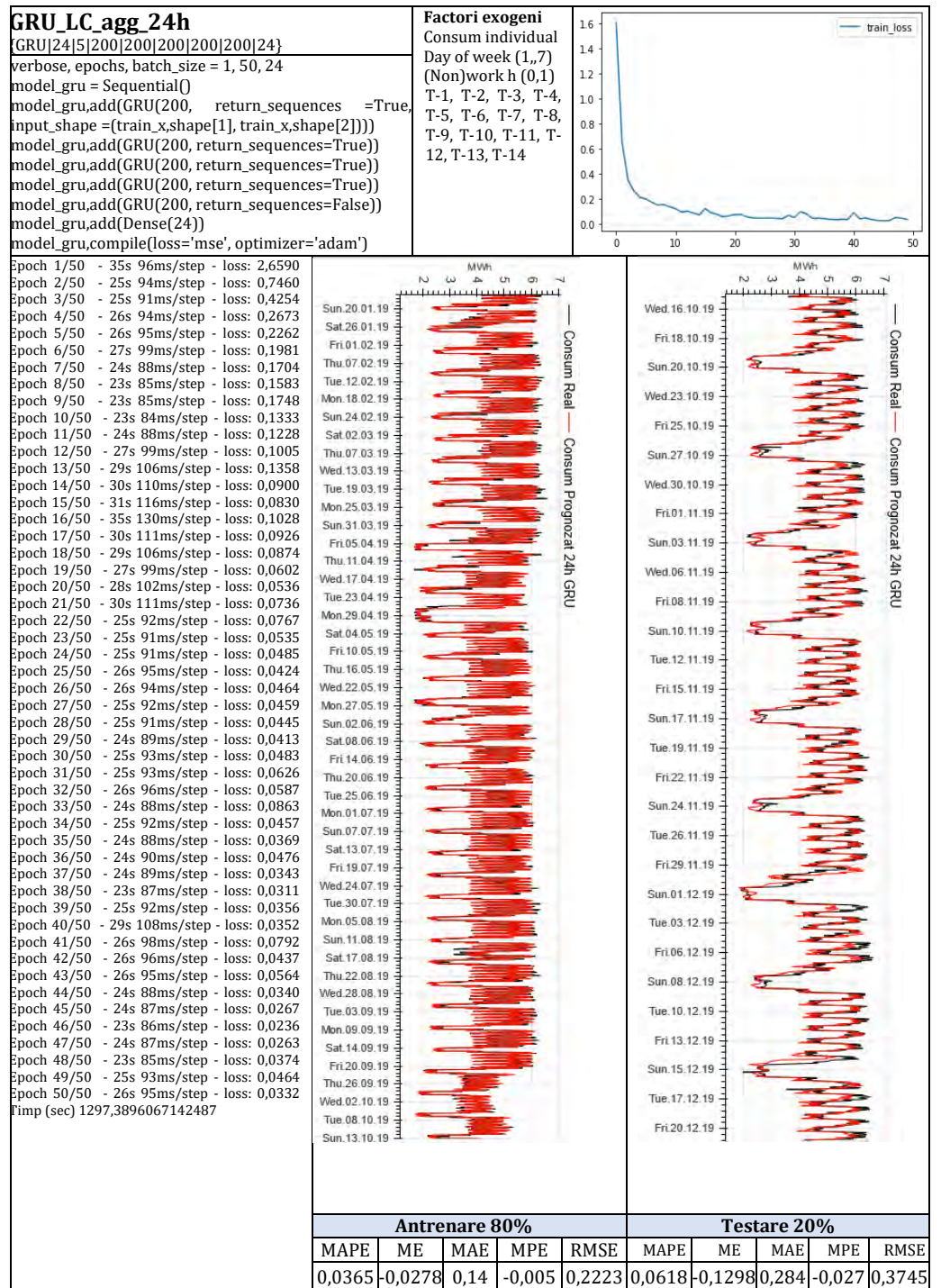
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4,
T-5, T-6, T-7, T-8,
T-9, T-10, T-11, T-
12, T-13, T-14



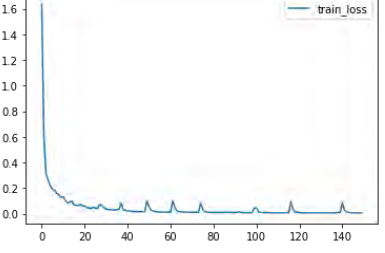

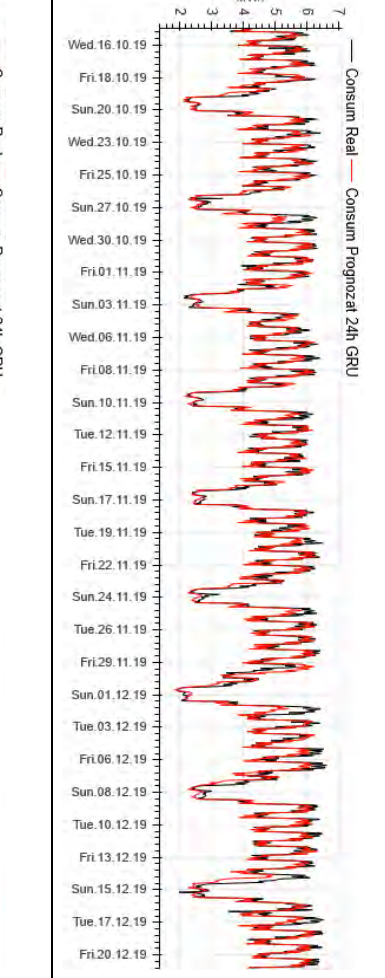
Epoch 1/30 - 34s 95ms/step - loss: 2,9767
Epoch 2/30 - 24s 89ms/step - loss: 0,7312
Epoch 3/30 - 26s 95ms/step - loss: 0,3287
Epoch 4/30 - 28s 103ms/step - loss: 0,2568
Epoch 5/30 - 27s 99ms/step - loss: 0,2077
Epoch 6/30 - 26s 98ms/step - loss: 0,2006
Epoch 7/30 - 26s 96ms/step - loss: 0,1587
Epoch 8/30 - 25s 91ms/step - loss: 0,1550
Epoch 9/30 - 25s 93ms/step - loss: 0,1468
Epoch 10/30 - 27s 101ms/step - loss: 0,1596
Epoch 11/30 - 27s 100ms/step - loss: 0,1135
Epoch 12/30 - 28s 102ms/step - loss: 0,0974
Epoch 13/30 - 28s 104ms/step - loss: 0,1004
Epoch 14/30 - 26s 96ms/step - loss: 0,0740
Epoch 15/30 - 28s 103ms/step - loss: 0,1036
Epoch 16/30 - 28s 103ms/step - loss: 0,0533
Epoch 17/30 - 27s 100ms/step - loss: 0,0515
Epoch 18/30 - 28s 102ms/step - loss: 0,1213
Epoch 19/30 - 27s 101ms/step - loss: 0,0696
Epoch 20/30 - 26s 95ms/step - loss: 0,0485
Epoch 21/30 - 25s 94ms/step - loss: 0,0493
Epoch 22/30 - 26s 95ms/step - loss: 0,0428
Epoch 23/30 - 27s 101ms/step - loss: 0,0401
Epoch 24/30 - 28s 105ms/step - loss: 0,0408
Epoch 25/30 - 29s 107ms/step - loss: 0,0680
Epoch 26/30 - 26s 95ms/step - loss: 0,0408
Epoch 27/30 - 26s 95ms/step - loss: 0,1454
Epoch 28/30 - 28s 103ms/step - loss: 0,0499
Epoch 29/30 - 28s 102ms/step - loss: 0,0370
Epoch 30/30 - 25s 94ms/step - loss: 0,0367
Timp (sec) 809,9506549835205



Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0449	0,1425	0,1832	0,034	0,249	0,0566	0,0909	0,2601	0,0203	0,352



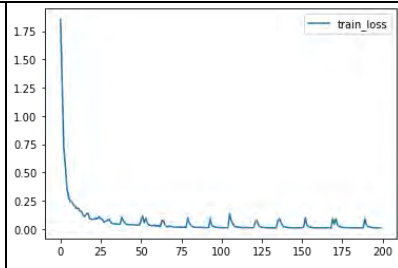
GRU_LC_agg_24h		Factori exogeni		train_loss							
<pre>GRU[24 5 200 200 200 200 24] verbose, epochs, batch_size = 1, 100, 24 model_gru = Sequential() model_gru.add(GRU(200, return_sequences =True, input_shape =(train_x.shape[1], train_x.shape[2]))) model_gru.add(GRU(200, return_sequences=True)) model_gru.add(GRU(200, return_sequences=True)) model_gru.add(GRU(200, return_sequences=True)) model_gru.add(GRU(200, return_sequences=False)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam')</pre>		<p>Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T- 12, T-13, T-14</p>									
<pre>Epoch 1/100 - 33s 90ms/step - loss: 2,8634 Epoch 2/100 - 23s 84ms/step - loss: 0,8944 Epoch 3/100 - 27s 99ms/step - loss: 0,6490 Epoch 4/100 - 26s 98ms/step - loss: 0,5159 Epoch 5/100 - 25s 92ms/step - loss: 0,3462 Epoch 6/100 - 26s 97ms/step - loss: 0,2755 Epoch 7/100 - 27s 98ms/step - loss: 0,2412 Epoch 8/100 - 26s 96ms/step - loss: 0,2273 Epoch 9/100 - 25s 91ms/step - loss: 0,1931 Epoch 10/100 - 27s 99ms/step - loss: 0,1857 Epoch 11/100 - 27s 101ms/step - loss: 0,18740 Epoch 12/100 - 27s 101ms/step - loss: 0,1403 Epoch 13/100 - 25s 93ms/step - loss: 0,1302 Epoch 14/100 - 25s 94ms/step - loss: 0,1512 Epoch 15/100 - 26s 97ms/step - loss: 0,1485 Epoch 16/100 - 23s 86ms/step - loss: 0,1224 Epoch 17/100 - 23s 86ms/step - loss: 0,1016 Epoch 18/100 - 24s 87ms/step - loss: 0,0840 Epoch 19/100 - 24s 88ms/step - loss: 0,1285 Epoch 20/100 - 24s 88ms/step - loss: 0,0854 Epoch 60/100 - 22s 82ms/step - loss: 0,0288 Epoch 61/100 - 22s 81ms/step - loss: 0,0288 Epoch 62/100 - 22s 82ms/step - loss: 0,0232 Epoch 63/100 - 22s 81ms/step - loss: 0,0273 Epoch 64/100 - 22s 82ms/step - loss: 0,0252 Epoch 65/100 - 22s 82ms/step - loss: 0,0222 Epoch 66/100 - 22s 82ms/step - loss: 0,0245 Epoch 67/100 - 22s 82ms/step - loss: 0,1527 Epoch 68/100 - 22s 82ms/step - loss: 0,0520 Epoch 69/100 - 22s 82ms/step - loss: 0,0274 Epoch 70/100 - 22s 82ms/step - loss: 0,0244 Epoch 71/100 - 22s 82ms/step - loss: 0,0225 Epoch 72/100 - 23s 83ms/step - loss: 0,0219 Epoch 73/100 - 22s 82ms/step - loss: 0,0182 Epoch 74/100 - 22s 82ms/step - loss: 0,0176 Epoch 75/100 - 22s 82ms/step - loss: 0,0176 Epoch 76/100 - 22s 82ms/step - loss: 0,0238 Epoch 77/100 - 22s 82ms/step - loss: 0,0988 Epoch 78/100 - 22s 82ms/step - loss: 0,0677 Epoch 79/100 - 22s 82ms/step - loss: 0,0347 Epoch 80/100 - 22s 82ms/step - loss: 0,0234 Epoch 81/100 - 22s 82ms/step - loss: 0,0194 Epoch 82/100 - 22s 82ms/step - loss: 0,0174 Epoch 83/100 - 22s 82ms/step - loss: 0,0157 Epoch 84/100 - 22s 82ms/step - loss: 0,0163 Epoch 85/100 - 22s 82ms/step - loss: 0,0148 Epoch 86/100 - 22s 83ms/step - loss: 0,0190 Epoch 87/100 - 22s 82ms/step - loss: 0,0242 Epoch 88/100 - 22s 82ms/step - loss: 0,0150 Epoch 89/100 - 22s 82ms/step - loss: 0,0139 Epoch 90/100 - 22s 82ms/step - loss: 0,0141 Epoch 91/100 - 22s 82ms/step - loss: 0,0138 Epoch 92/100 - 22s 82ms/step - loss: 0,0130 Epoch 93/100 - 22s 82ms/step - loss: 0,0140 Epoch 94/100 - 22s 82ms/step - loss: 0,0323 Epoch 95/100 - 22s 82ms/step - loss: 0,1341 Epoch 96/100 - 22s 81ms/step - loss: 0,0538 Epoch 97/100 - 22s 82ms/step - loss: 0,0414 Epoch 98/100 - 22s 81ms/step - loss: 0,0215 Epoch 99/100 - 22s 81ms/step - loss: 0,0213 Epoch 100/100 - 22s 81ms/step - loss: 0,0169</pre>											
		Antrenare 80%		Testare 20%							
		MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
		0,0236	-0,0025	0,096	0,0001	0,1277	0,0566	-0,0817	0,266	-0,0176	0,361
<pre>limp (sec) 2356,3253350257874</pre>											

GRU_LC_agg_24h	Factori exogeni																															
<pre>GRU[24]5[200]200[200]200[200]24} verbose, epochs, batch_size = 1, 150, 24 model_gru = Sequential() model_gru.add(GRU(200, return_sequences =True, input_shape =(train_x.shape[1], train_x.shape[2]))) model_gru.add(GRU(200, return_sequences=True)) model_gru.add(GRU(200, return_sequences=True)) model_gru.add(GRU(200, return_sequences=True)) model_gru.add(GRU(200, return_sequences=False)) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam')</pre>	<p>Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T- 12, T-13, T-14</p>																															
<pre>Epoch 1/150 - 32s 88ms/step - loss: 2,8316 Epoch 2/150 - 24s 88ms/step - loss: 0,7309 Epoch 3/150 - 25s 92ms/step - loss: 0,3418 Epoch 4/150 - 25s 91ms/step - loss: 0,2796 Epoch 5/150 - 25s 93ms/step - loss: 0,2206 Epoch 6/150 - 25s 91ms/step - loss: 0,2012 Epoch 7/150 - 24s 89ms/step - loss: 0,1808 Epoch 8/150 - 27s 99ms/step - loss: 0,1727 Epoch 9/150 - 27s 98ms/step - loss: 0,1560 Epoch 10/150 - 25s 92ms/step - loss: 0,1293 Epoch 11/150 - 26s 96ms/step - loss: 0,1225 Epoch 12/150 - 26s 95ms/step - loss: 0,1071 Epoch 13/150 - 26s 97ms/step - loss: 0,0892 Epoch 14/150 - 27s 99ms/step - loss: 0,0840 Epoch 15/150 - 26s 87ms/step - loss: 0,0779 Epoch 17/150 - 25s 93ms/step - loss: 0,0719 Epoch 18/150 - 25s 93ms/step - loss: 0,0770 Epoch 19/150 - 23s 85ms/step - loss: 0,0650 Epoch 20/150 - 24s 87ms/step - loss: 0,0640 Epoch 113/150 - 24s 88ms/step - loss: 0,007 Epoch 114/150 - 24s 89ms/step - loss: 0,007 Epoch 115/150 - 25s 93ms/step - loss: 0,008 Epoch 116/150 - 26s 94ms/step - loss: 0,007 Epoch 117/150 - 24s 88ms/step - loss: 0,110 Epoch 118/150 - 27s 99ms/step - loss: 0,044 Epoch 119/150 - 28s 103ms/step - loss: 0,015 Epoch 120/150 - 26s 96ms/step - loss: 0,011 Epoch 121/150 - 24s 90ms/step - loss: 0,009 Epoch 122/150 - 24s 89ms/step - loss: 0,007 Epoch 123/150 - 23s 85ms/step - loss: 0,007 Epoch 124/150 - 23s 86ms/step - loss: 0,007 Epoch 125/150 - 23s 86ms/step - loss: 0,007 Epoch 126/150 - 23s 85ms/step - loss: 0,006 Epoch 127/150 - 23s 85ms/step - loss: 0,006 Epoch 128/150 - 23s 86ms/step - loss: 0,006 Epoch 129/150 - 23s 86ms/step - loss: 0,006 Epoch 130/150 - 23s 86ms/step - loss: 0,007 Epoch 131/150 - 23s 85ms/step - loss: 0,007 Epoch 132/150 - 23s 85ms/step - loss: 0,007 Epoch 133/150 - 23s 86ms/step - loss: 0,008 Epoch 134/150 - 24s 87ms/step - loss: 0,007 Epoch 135/150 - 23s 86ms/step - loss: 0,007 Epoch 136/150 - 23s 86ms/step - loss: 0,006 Epoch 137/150 - 23s 85ms/step - loss: 0,007 Epoch 138/150 - 23s 86ms/step - loss: 0,007 Epoch 139/150 - 23s 86ms/step - loss: 0,011 Epoch 140/150 - 23s 85ms/step - loss: 0,008 Epoch 141/150 - 23s 85ms/step - loss: 0,080 Epoch 142/150 - 23s 85ms/step - loss: 0,049 Epoch 143/150 - 23s 86ms/step - loss: 0,015 Epoch 144/150 - 23s 85ms/step - loss: 0,010 Epoch 145/150 - 23s 86ms/step - loss: 0,008 Epoch 146/150 - 23s 85ms/step - loss: 0,007 Epoch 147/150 - 23s 85ms/step - loss: 0,006 Epoch 148/150 - 23s 85ms/step - loss: 0,006 Epoch 149/150 - 23s 86ms/step - loss: 0,006 Epoch 150/150 - 23s 86ms/step - loss: 0,005 Timp (sec) 3555,198965549469</pre>																																
	<table border="1"> <thead> <tr> <th colspan="5">Antrenare 80%</th> <th colspan="5">Testare 20%</th> </tr> <tr> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> </tr> </thead> <tbody> <tr> <td>0,0147</td> <td>-0,0072</td> <td>0,059</td> <td>-0,0013</td> <td>0,0763</td> <td>0,0581</td> <td>-0,1142</td> <td>0,275</td> <td>-0,022</td> <td>0,3764</td> </tr> </tbody> </table>	Antrenare 80%					Testare 20%					MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE	0,0147	-0,0072	0,059	-0,0013	0,0763	0,0581	-0,1142	0,275	-0,022	0,3764	
Antrenare 80%					Testare 20%																											
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE																							
0,0147	-0,0072	0,059	-0,0013	0,0763	0,0581	-0,1142	0,275	-0,022	0,3764																							

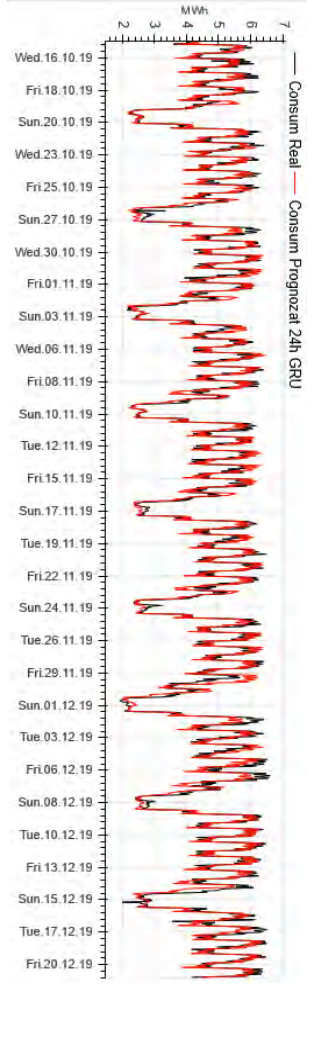
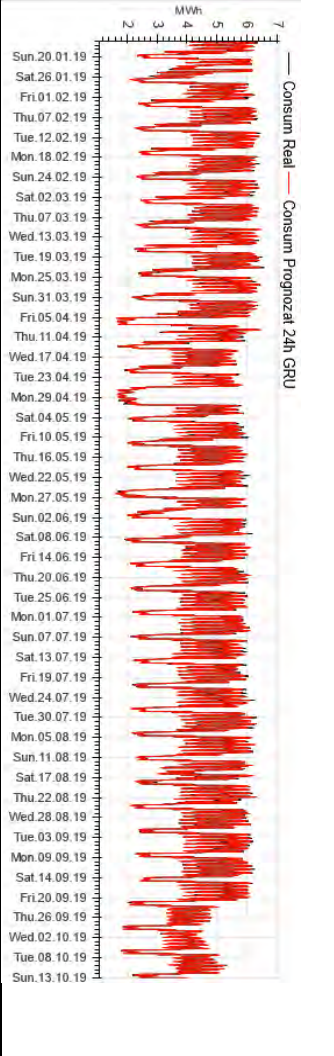
GRU_LC_agg_24h

```
{GRU[24|5|200|200|200|200|24]}
verbose, epochs, batch_size = 1, 200, 24
model_gru = Sequential()
model_gru.add(GRU(200, return_sequences = True,
input_shape = (train_x.shape[1], train_x.shape[2])))
model_gru.add(GRU(200, return_sequences=True))
model_gru.add(GRU(200, return_sequences=True))
model_gru.add(GRU(200, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
```

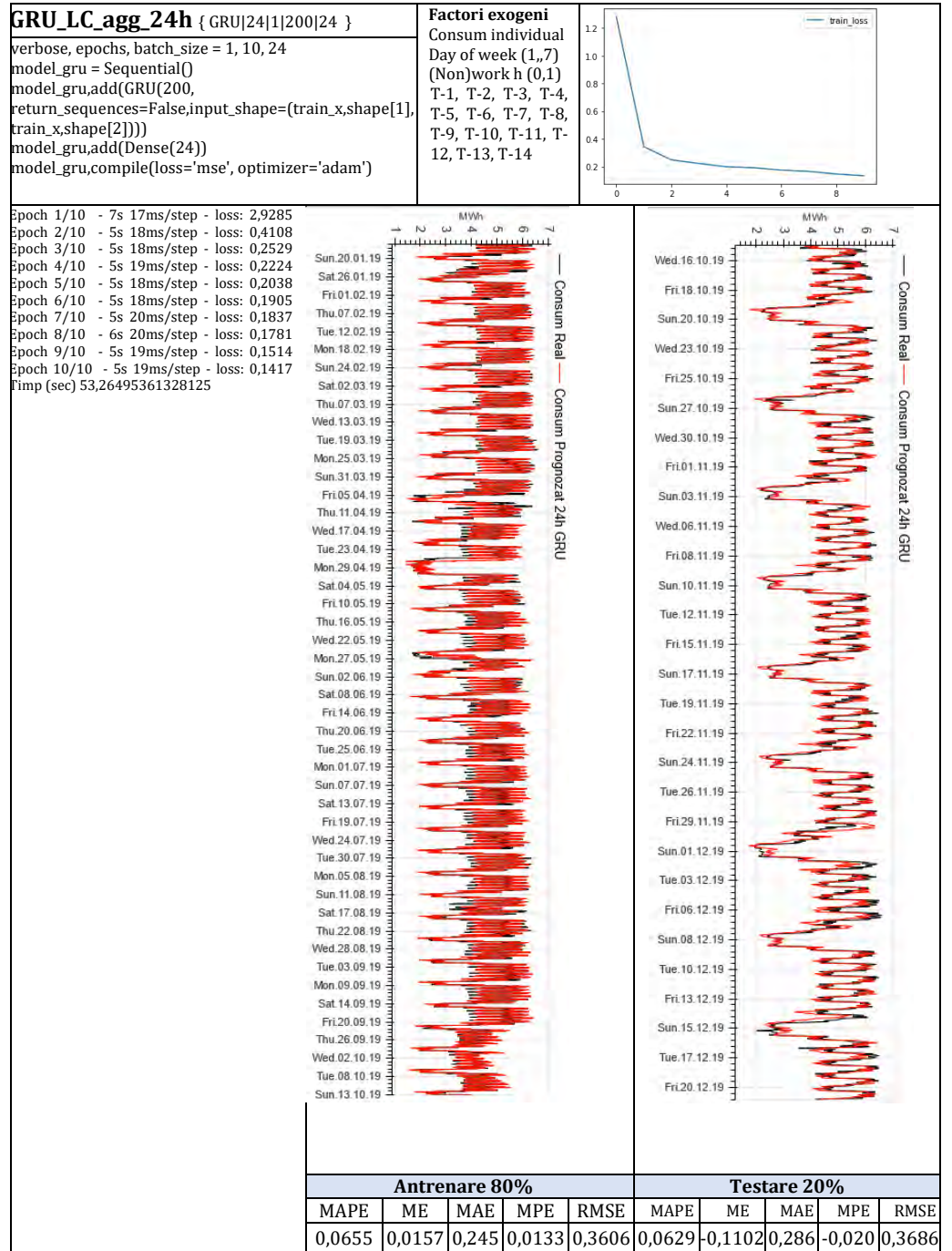
Factori exogeni
Consum individual
Day of week (1,,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4,
T-5, T-6, T-7, T-8,
T-9, T-10, T-11, T-
12, T-13, T-14



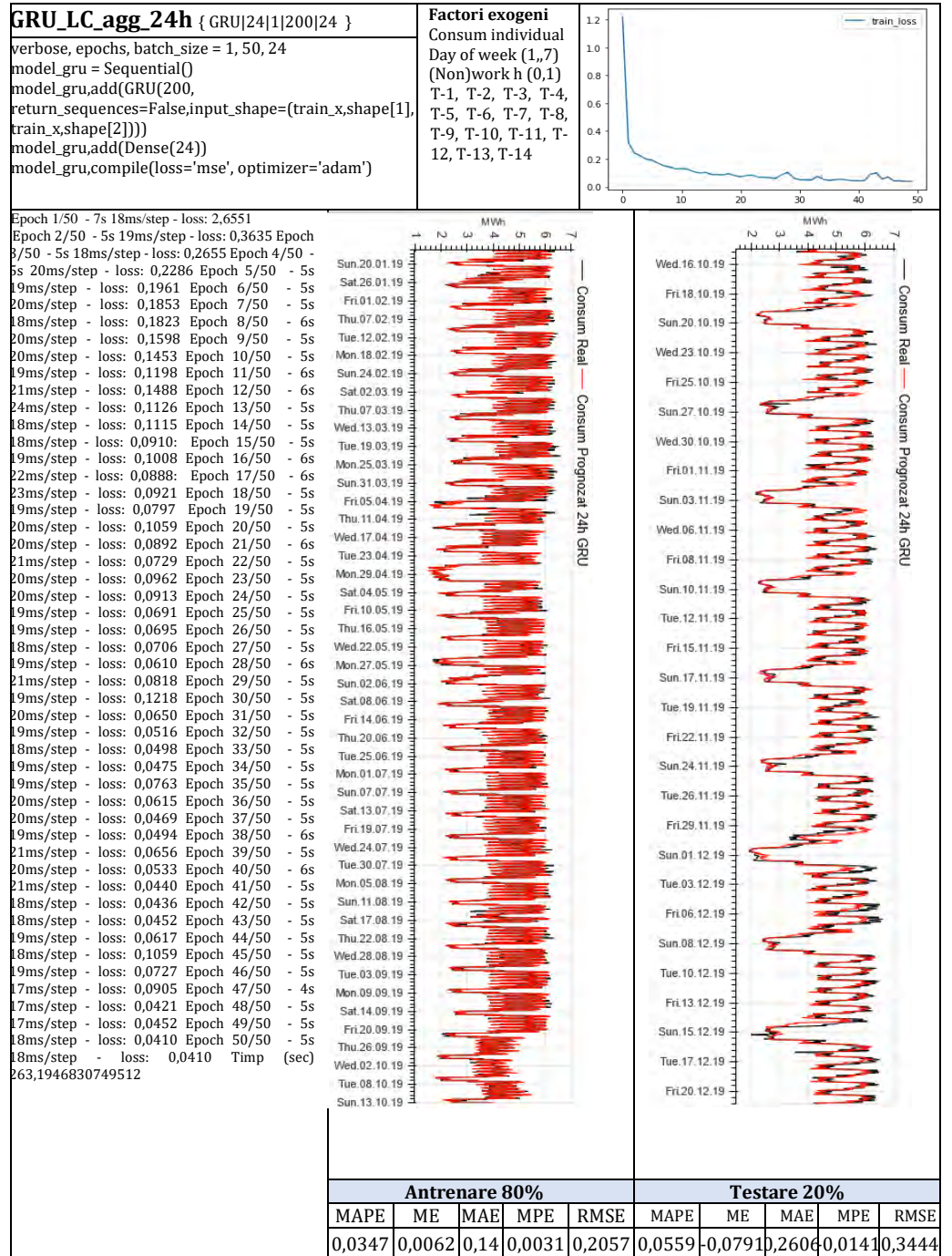
Epoch 1/200 - 36s 100ms/step - loss: 2,8022
Epoch 2/200 - 27s 100ms/step - loss: 1,4913
Epoch 3/200 - 27s 100ms/step - loss: 0,7915
Epoch 4/200 - 28s 103ms/step - loss: 0,6064
Epoch 5/200 - 25s 94ms/step - loss: 0,4018
Epoch 6/200 - 24s 89ms/step - loss: 0,2986
Epoch 7/200 - 25s 94ms/step - loss: 0,2518
Epoch 8/200 - 26s 98ms/step - loss: 0,2355
Epoch 9/200 - 26s 96ms/step - loss: 0,2171
Epoch 10/200 - 26s 95ms/step - loss: 0,1855
Epoch 11/200 - 27s 100ms/step - loss: 0,1831
Epoch 12/200 - 26s 96ms/step - loss: 0,1917
Epoch 13/200 - 26s 95ms/step - loss: 0,1620
Epoch 14/200 - 27s 101ms/step - loss: 0,1464
Epoch 15/200 - 28s 102ms/step - loss: 0,1371
Epoch 16/200 - 28s 105ms/step - loss: 0,1226
Epoch 17/200 - 25s 93ms/step - loss: 0,1326
Epoch 18/200 - 27s 101ms/step - loss: 0,13830
Epoch 19/200 - 26s 97ms/step - loss: 0,1077
Epoch 20/200 - 27s 100ms/step - loss: 0,0932
.....
Epoch 162/200 - 26s 95ms/step - loss: 0,0109
Epoch 163/200 - 24s 90ms/step - loss: 0,0102
Epoch 164/200 - 24s 90ms/step - loss: 0,0099
Epoch 165/200 - 24s 89ms/step - loss: 0,0105
Epoch 166/200 - 24s 90ms/step - loss: 0,0135
Epoch 167/200 - 24s 90ms/step - loss: 0,0118
Epoch 168/200 - 24s 89ms/step - loss: 0,0105
Epoch 169/200 - 25s 91ms/step - loss: 0,0092
Epoch 170/200 - 25s 91ms/step - loss: 0,0775
Epoch 171/200 - 24s 90ms/step - loss: 0,0511
Epoch 172/200 - 25s 91ms/step - loss: 0,1311
Epoch 173/200 - 25s 92ms/step - loss: 0,0397
Epoch 174/200 - 25s 90ms/step - loss: 0,0302
Epoch 175/200 - 24s 90ms/step - loss: 0,0182
Epoch 176/200 - 24s 90ms/step - loss: 0,0152
Epoch 177/200 - 24s 90ms/step - loss: 0,0135
Epoch 178/200 - 25s 91ms/step - loss: 0,0122
Epoch 179/200 - 24s 89ms/step - loss: 0,0126
Epoch 180/200 - 24s 90ms/step - loss: 0,0111
Epoch 181/200 - 24s 90ms/step - loss: 0,0123
Epoch 182/200 - 25s 91ms/step - loss: 0,0105
Epoch 183/200 - 24s 90ms/step - loss: 0,0108
Epoch 184/200 - 24s 89ms/step - loss: 0,0142
Epoch 185/200 - 24s 90ms/step - loss: 0,0102
Epoch 186/200 - 24s 90ms/step - loss: 0,0103
Epoch 187/200 - 24s 89ms/step - loss: 0,0098
Epoch 188/200 - 25s 91ms/step - loss: 0,0103
Epoch 189/200 - 24s 89ms/step - loss: 0,0104
Epoch 190/200 - 24s 90ms/step - loss: 0,0662
Epoch 191/200 - 24s 89ms/step - loss: 0,0400
Epoch 192/200 - 24s 89ms/step - loss: 0,0291
Epoch 193/200 - 24s 90ms/step - loss: 0,0164
Epoch 194/200 - 24s 89ms/step - loss: 0,0152
Epoch 195/200 - 24s 88ms/step - loss: 0,0121
Epoch 196/200 - 24s 89ms/step - loss: 0,0109
Epoch 197/200 - 24s 90ms/step - loss: 0,0102
Epoch 198/200 - 24s 89ms/step - loss: 0,0099
Epoch 199/200 - 24s 89ms/step - loss: 0,0100
Epoch 200/200 - 24s 90ms/step - loss: 0,0092
Timp (sec) 4918,575264215469

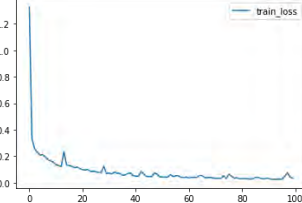
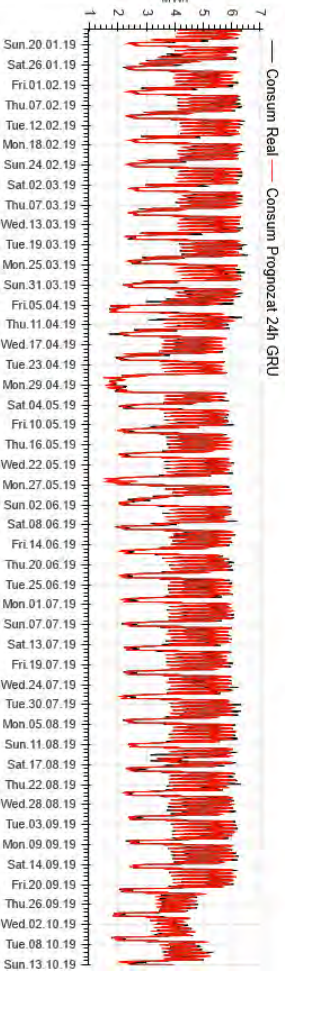
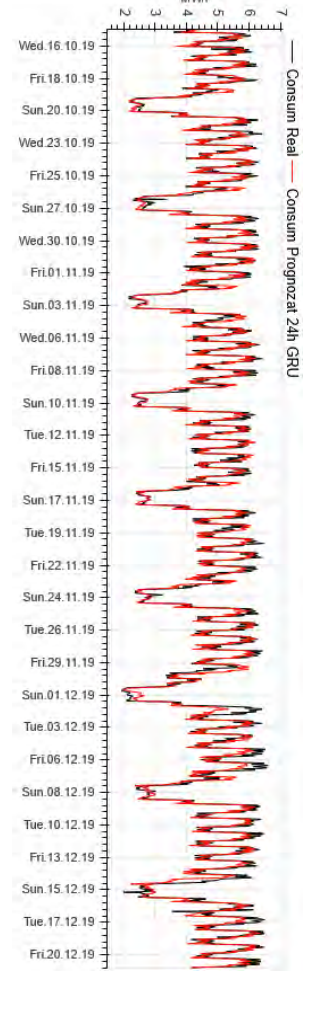


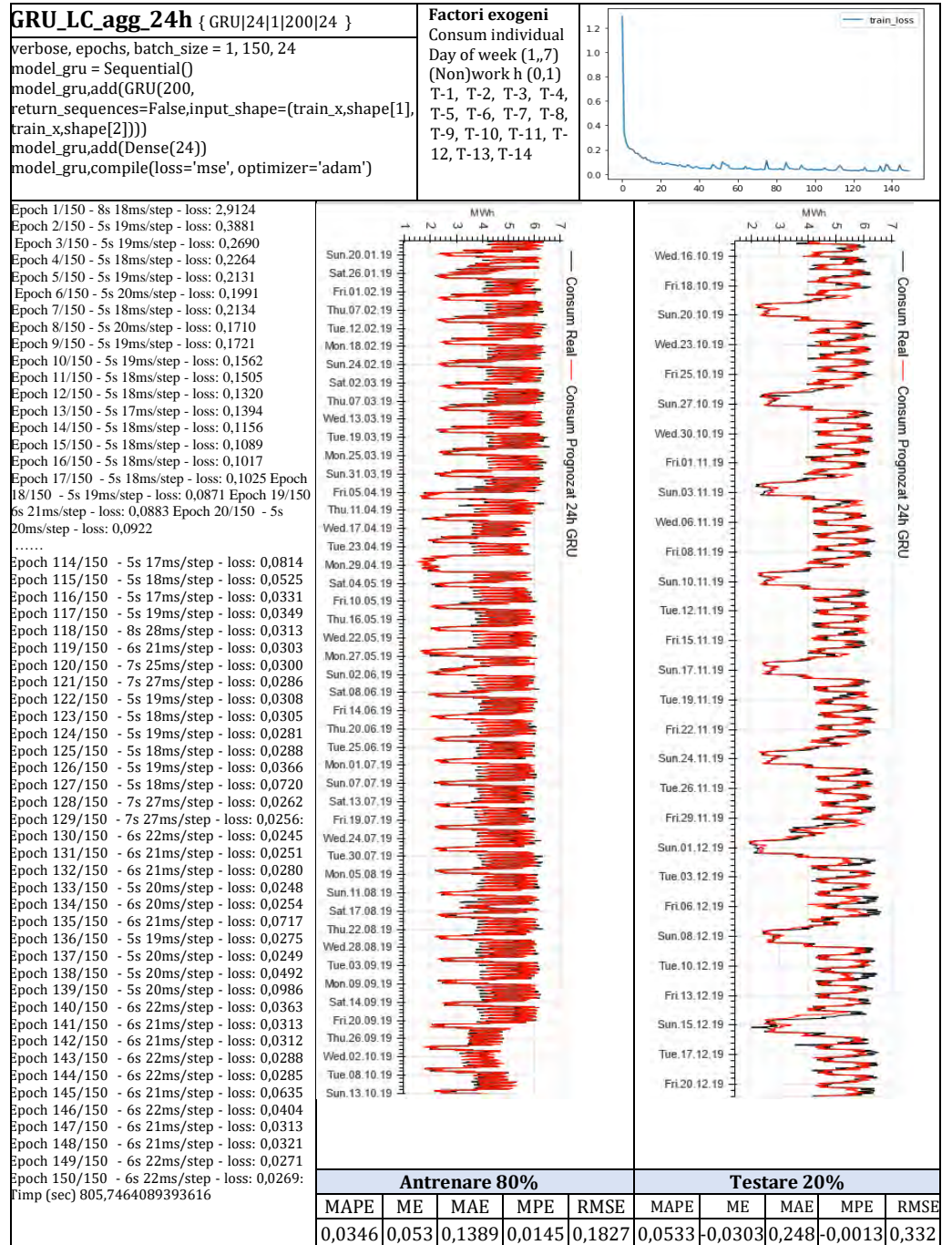
Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,018	-0,0036	0,073	-0,0007	0,0946	0,0588	-0,075	0,267	-0,015	0,3519

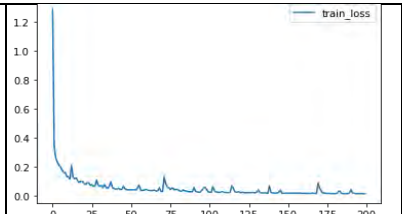
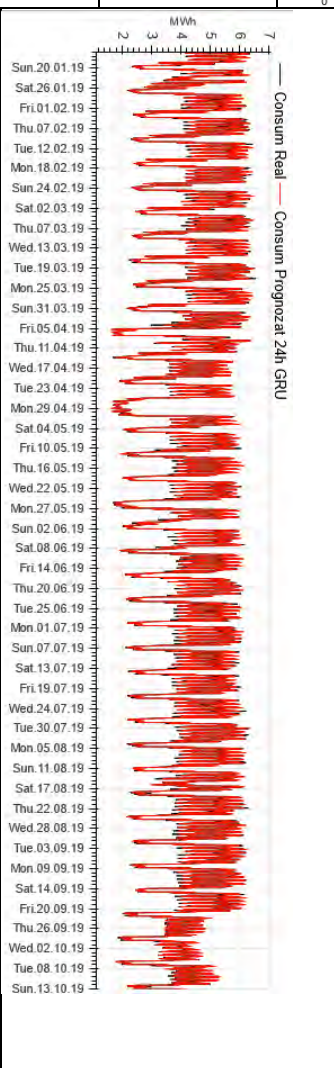
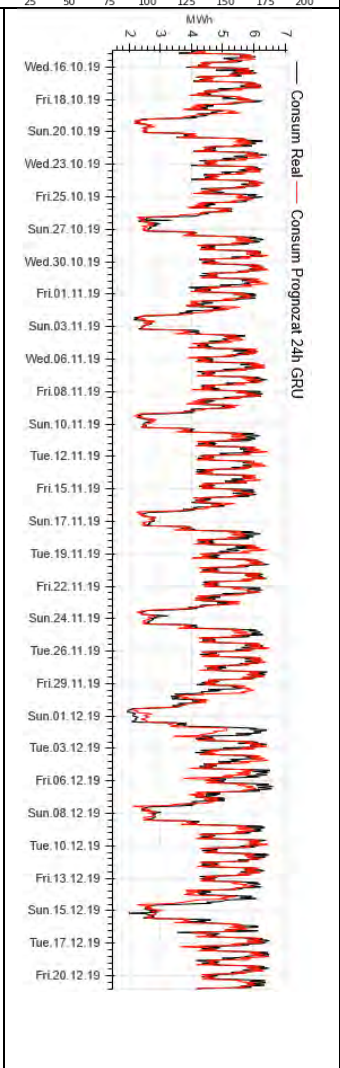


GRU_IC_agg_24h { GRU[24][1][200][24] }		Factori exogeni							
verbose, epochs, batch_size = 1, 30, 24 model_gru = Sequential() model_gru.add(GRU(200, return_sequences=False,input_shape=(train_x.shape[1], train_x.shape[2]))) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam')		Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T- 12, T-13, T-14							
Epoch 1/30 - 7s 17ms/step - loss: 3,3492 Epoch 2/30 - 5s 18ms/step - loss: 0,4140 Epoch 3/30 - 5s 17ms/step - loss: 0,2610 Epoch 4/30 - 5s 17ms/step - loss: 0,2419 Epoch 5/30 - 5s 17ms/step - loss: 0,2066 Epoch 6/30 - 5s 17ms/step - loss: 0,1963 Epoch 7/30 - 5s 17ms/step - loss: 0,1832 Epoch 8/30 - 5s 17ms/step - loss: 0,1635 Epoch 9/30 - 6s 21ms/step - loss: 0,1627 Epoch 10/30 - 5s 18ms/step - loss: 0,1367 Epoch 11/30 - 5s 17ms/step - loss: 0,1297 Epoch 12/30 - 4s 16ms/step - loss: 0,1290 Epoch 13/30 - 6s 22ms/step - loss: 0,1163 Epoch 14/30 - 6s 20ms/step - loss: 0,1032 Epoch 15/30 - 4s 16ms/step - loss: 0,1176 Epoch 16/30 - 5s 18ms/step - loss: 0,1159 Epoch 17/30 - 5s 19ms/step - loss: 0,0998 Epoch 18/30 - 5s 18ms/step - loss: 0,0937 Epoch 19/30 - 5s 19ms/step - loss: 0,1078 Epoch 20/30 - 5s 18ms/step - loss: 0,0836 Epoch 21/30 - 5s 18ms/step - loss: 0,1192 Epoch 22/30 - 5s 18ms/step - loss: 0,0717 Epoch 23/30 - 5s 17ms/step - loss: 0,0774 Epoch 24/30 - 5s 18ms/step - loss: 0,0861 Epoch 25/30 - 5s 19ms/step - loss: 0,0792 Epoch 26/30 - 5s 18ms/step - loss: 0,0681 Epoch 27/30 - 5s 18ms/step - loss: 0,0743 Epoch 28/30 - 5s 18ms/step - loss: 0,0655 Epoch 29/30 - 5s 17ms/step - loss: 0,0683 Epoch 30/30 - 5s 17ms/step - loss: 0,0722 Timp (sec) 148,17795252799988									
		Antrenare 80%	Testare 20%						
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,0469	-0,0369	0,183	-0,0073	0,2791	0,0621	-0,1593	0,287	-0,0344	0,3802



GRU_LC_agg_24h { GRU 24 1 200 24 } verbose, epochs, batch_size = 1, 100, 24 model_gru = Sequential() model_gru.add(GRU(200, return_sequences=False,input_shape=(train_x.shape[1], train_x.shape[2]))) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam')	Factori exogeni Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T- 12, T-13, T-14									
Epoch 1/100 - 7s 19ms/step - loss: 3,0726 Epoch 2/100 - 5s 18ms/step - loss: 0,3793 Epoch 3/100 - 5s 18ms/step - loss: 0,2632 Epoch 4/100 - 5s 20ms/step - loss: 0,2338 Epoch 5/100 - 5s 20ms/step - loss: 0,2136 Epoch 6/100 - 5s 19ms/step - loss: 0,2469 Epoch 7/100 - 6s 22ms/step - loss: 0,1906 Epoch 8/100 - 5s 17ms/step - loss: 0,1827 Epoch 9/100 - 5s 19ms/step - loss: 0,1647 Epoch 10/100 - 5s 19ms/step - loss: 0,1706 Epoch 11/100 - 5s 18ms/step - loss: 0,1416 Epoch 12/100 - 5s 17ms/step - loss: 0,1264 Epoch 13/100 - 5s 18ms/step - loss: 0,1243 Epoch 14/100 - 5s 17ms/step - loss: 0,3415 Epoch 15/100 - 5s 19ms/step - loss: 0,1423 Epoch 16/100 - 5s 18ms/step - loss: 0,1308 Epoch 17/100 - 6s 22ms/step - loss: 0,1240 Epoch 18/100 - 7s 24ms/step - loss: 0,1155 Epoch 19/100 - 5s 18ms/step - loss: 0,1281 Epoch 20/100 - 5s 19ms/step - loss: 0,1157 Epoch 21/100 - 5s 20ms/step - loss: 0,0952 Epoch 22/100 - 5s 18ms/step - loss: 0,1078 Epoch 23/100 - 5s 19ms/step - loss: 0,1070 Epoch 24/100 - 5s 20ms/step - loss: 0,0837 Epoch 25/100 - 5s 19ms/step - loss: 0,0828 Epoch 26/100 - 5s 20ms/step - loss: 0,0843 Epoch 27/100 - 6s 21ms/step - loss: 0,0814 Epoch 28/100 - 5s 18ms/step - loss: 0,0783 Epoch 29/100 - 5s 17ms/step - loss: 0,1362 Epoch 30/100 - 5s 18ms/step - loss: 0,0717 Epoch 70/100 - 5s 19ms/step - loss: 0,0381 Epoch 71/100 - 5s 18ms/step - loss: 0,0364 Epoch 72/100 - 5s 19ms/step - loss: 0,0359 Epoch 73/100 - 5s 19ms/step - loss: 0,0330 Epoch 74/100 - 5s 19ms/step - loss: 0,0517 Epoch 75/100 - 5s 18ms/step - loss: 0,0346 Epoch 76/100 - 5s 18ms/step - loss: 0,0358 Epoch 77/100 - 5s 19ms/step - loss: 0,0535 Epoch 78/100 - 5s 19ms/step - loss: 0,0384 Epoch 79/100 - 5s 19ms/step - loss: 0,0371 Epoch 80/100 - 5s 19ms/step - loss: 0,0296 Epoch 81/100 - 5s 19ms/step - loss: 0,0319 Epoch 82/100 - 5s 18ms/step - loss: 0,0308 Epoch 83/100 - 5s 19ms/step - loss: 0,0307 Epoch 84/100 - 5s 17ms/step - loss: 0,0286 Epoch 85/100 - 5s 17ms/step - loss: 0,0289 Epoch 86/100 - 5s 18ms/step - loss: 0,0323 Epoch 87/100 - 5s 20ms/step - loss: 0,0527 Epoch 88/100 - 5s 18ms/step - loss: 0,0329 Epoch 89/100 - 5s 19ms/step - loss: 0,0299 Epoch 90/100 - 5s 19ms/step - loss: 0,0294 Epoch 91/100 - 5s 19ms/step - loss: 0,0361 Epoch 92/100 - 5s 18ms/step - loss: 0,0265 Epoch 93/100 - 5s 17ms/step - loss: 0,0272 Epoch 94/100 - 5s 17ms/step - loss: 0,0261 Epoch 95/100 - 5s 17ms/step - loss: 0,0284 Epoch 96/100 - 5s 18ms/step - loss: 0,0272 Epoch 97/100 - 5s 18ms/step - loss: 0,0290 Epoch 98/100 - 5s 18ms/step - loss: 0,1062 Epoch 99/100 - 5s 17ms/step - loss: 0,0451 Epoch 100/100 - 5s 17ms/step - loss: 0,0347 Timp (sec) 523,2971155643463										
	Antrenare 80%					Testare 20%				
	MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
	0,0329	0,0073	0,132	0,0018	0,187	0,0532	-0,0828	0,248	-0,013	0,3307



GRU_LC_agg_24h {GRU 24 1 200 24 } verbose, epochs, batch_size = 1, 200, 24 model_gru = Sequential() model_gru.add(GRU(200, return_sequences=False,input_shape=(train_x.shape[1], train_x.shape[2]))) model_gru.add(Dense(24)) model_gru.compile(loss='mse', optimizer='adam')	Factori exogeni Consum individual Day of week (1,,7) (Non)work h (0,1) T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-9, T-10, T-11, T- 12, T-13, T-14																															
<p>Epoch 1/200 - 7s 16ms/step - loss: 2,9173 Epoch 2/200 - 5s 17ms/step - loss: 0,3947 Epoch 3/200 - 4s 15ms/step - loss: 0,2671 Epoch 4/200 - 4s 15ms/step - loss: 0,2328 Epoch 5/200 - 4s 16ms/step - loss: 0,2169 Epoch 6/200 - 4s 16ms/step - loss: 0,2065 Epoch 7/200 - 4s 16ms/step - loss: 0,1891 Epoch 8/200 - 4s 16ms/step - loss: 0,1689 Epoch 9/200 - 4s 16ms/step - loss: 0,1866 Epoch 10/200 - 4s 16ms/step - loss: 0,1403 Epoch 148/200 - 4s 16ms/step - loss: 0,0173 Epoch 149/200 - 4s 16ms/step - loss: 0,0193 Epoch 150/200 - 4s 16ms/step - loss: 0,0185 Epoch 151/200 - 4s 16ms/step - loss: 0,0178 Epoch 152/200 - 4s 16ms/step - loss: 0,0186 Epoch 153/200 - 4s 16ms/step - loss: 0,0177 Epoch 154/200 - 4s 16ms/step - loss: 0,0185 Epoch 155/200 - 4s 16ms/step - loss: 0,0187 Epoch 156/200 - 4s 16ms/step - loss: 0,0205 Epoch 157/200 - 4s 16ms/step - loss: 0,0188 Epoch 158/200 - 4s 16ms/step - loss: 0,0178 Epoch 159/200 - 4s 16ms/step - loss: 0,0172 Epoch 160/200 - 4s 16ms/step - loss: 0,0185 Epoch 161/200 - 4s 16ms/step - loss: 0,0173 Epoch 162/200 - 5s 17ms/step - loss: 0,0182 Epoch 163/200 - 5s 17ms/step - loss: 0,0180 Epoch 164/200 - 4s 16ms/step - loss: 0,0175 Epoch 165/200 - 4s 15ms/step - loss: 0,0163 Epoch 166/200 - 5s 17ms/step - loss: 0,0178 Epoch 167/200 - 5s 17ms/step - loss: 0,0200 Epoch 168/200 - 4s 17ms/step - loss: 0,0168 Epoch 169/200 - 4s 16ms/step - loss: 0,0168 Epoch 170/200 - 5s 17ms/step - loss: 0,0560 Epoch 171/200 - 4s 16ms/step - loss: 0,0593 Epoch 172/200 - 5s 17ms/step - loss: 0,0406 Epoch 173/200 - 4s 16ms/step - loss: 0,0214 Epoch 174/200 - 5s 17ms/step - loss: 0,0187 Epoch 175/200 - 4s 15ms/step - loss: 0,0179 Epoch 176/200 - 4s 15ms/step - loss: 0,0177 Epoch 177/200 - 4s 16ms/step - loss: 0,0163 Epoch 178/200 - 4s 15ms/step - loss: 0,0162 Epoch 179/200 - 5s 17ms/step - loss: 0,0166 Epoch 180/200 - 5s 17ms/step - loss: 0,0157 Epoch 181/200 - 4s 16ms/step - loss: 0,0152 Epoch 182/200 - 5s 17ms/step - loss: 0,0164 Epoch 183/200 - 4s 16ms/step - loss: 0,0172 Epoch 184/200 - 4s 17ms/step - loss: 0,0451 Epoch 185/200 - 4s 16ms/step - loss: 0,0162 Epoch 186/200 - 5s 17ms/step - loss: 0,0151 Epoch 187/200 - 5s 17ms/step - loss: 0,0146 Epoch 188/200 - 5s 17ms/step - loss: 0,0154 Epoch 189/200 - 5s 19ms/step - loss: 0,0171 Epoch 190/200 - 4s 16ms/step - loss: 0,0185 Epoch 191/200 - 5s 17ms/step - loss: 0,0561 Epoch 192/200 - 5s 17ms/step - loss: 0,0219 Epoch 193/200 - 5s 18ms/step - loss: 0,0174 Epoch 194/200 - 4s 16ms/step - loss: 0,0164 Epoch 195/200 - 4s 16ms/step - loss: 0,0157 Epoch 196/200 - 5s 18ms/step - loss: 0,0161 Epoch 197/200 - 4s 16ms/step - loss: 0,0153 Epoch 198/200 - 4s 16ms/step - loss: 0,0153 Epoch 199/200 - 4s 16ms/step - loss: 0,0144 Epoch 200/200 - 4s 16ms/step - loss: 0,0150 Timp (sec) 362,0377173423767</p>																																
	<table border="1" style="width:100%; text-align:center;"> <thead> <tr> <th colspan="5">Antrenare 80%</th> <th colspan="5">Testare 20%</th> </tr> <tr> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> <th>MAPE</th> <th>ME</th> <th>MAE</th> <th>MPE</th> <th>RMSE</th> </tr> </thead> <tbody> <tr> <td>0,0254</td> <td>0,054</td> <td>0,104</td> <td>0,014</td> <td>0,1352</td> <td>0,0578</td> <td>-0,043</td> <td>0,27</td> <td>-0,006</td> <td>0,361</td> </tr> </tbody> </table>		Antrenare 80%					Testare 20%					MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE	0,0254	0,054	0,104	0,014	0,1352	0,0578	-0,043	0,27	-0,006	0,361
Antrenare 80%					Testare 20%																											
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE																							
0,0254	0,054	0,104	0,014	0,1352	0,0578	-0,043	0,27	-0,006	0,361																							

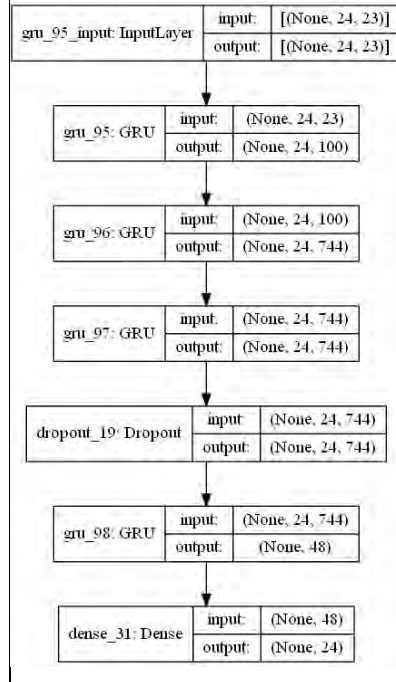
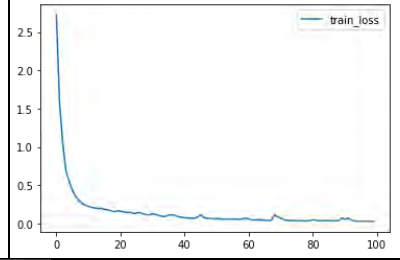
GRU_LC_agg_24h {GRU|24|100|744|744|48 }

```

verbose, epochs, batch_size = 1, 100, 24
model_gru = Sequential()
model_gru.add(GRU(100,return_sequences=True,
input_shape=(train_x.shape[1], train_x.shape[2])))
model_gru.add(GRU(744, return_sequences=True))
model_gru.add(GRU(744, return_sequences=True))
model_gru.add(Dropout(0,5))
model_gru.add(GRU(48, return_sequences=False))
model_gru.add(Dense(24))
model_gru.compile(loss='mse', optimizer='adam')
    
```

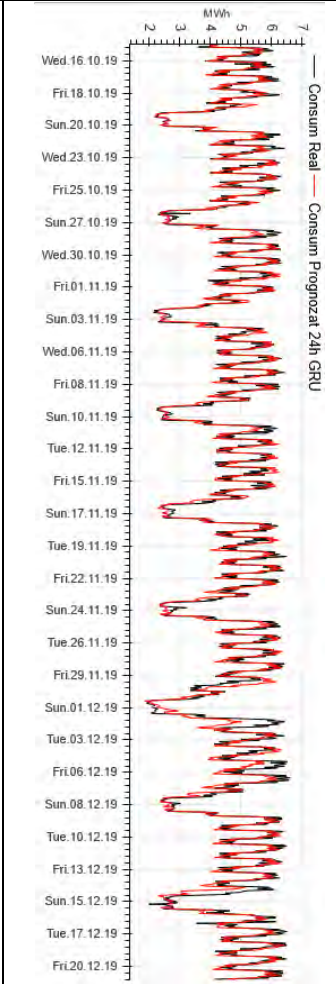
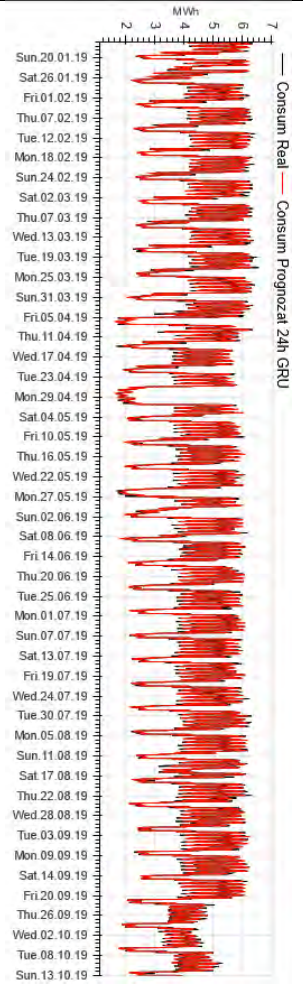
Factori exogeni

Consum individual
Day of week (1,7)
(Non)work h (0,1)
T-1, T-2, T-3, T-4,
T-5, T-6, T-7, T-8,
T-9, T-10, T-11, T-12,
T-13, T-14

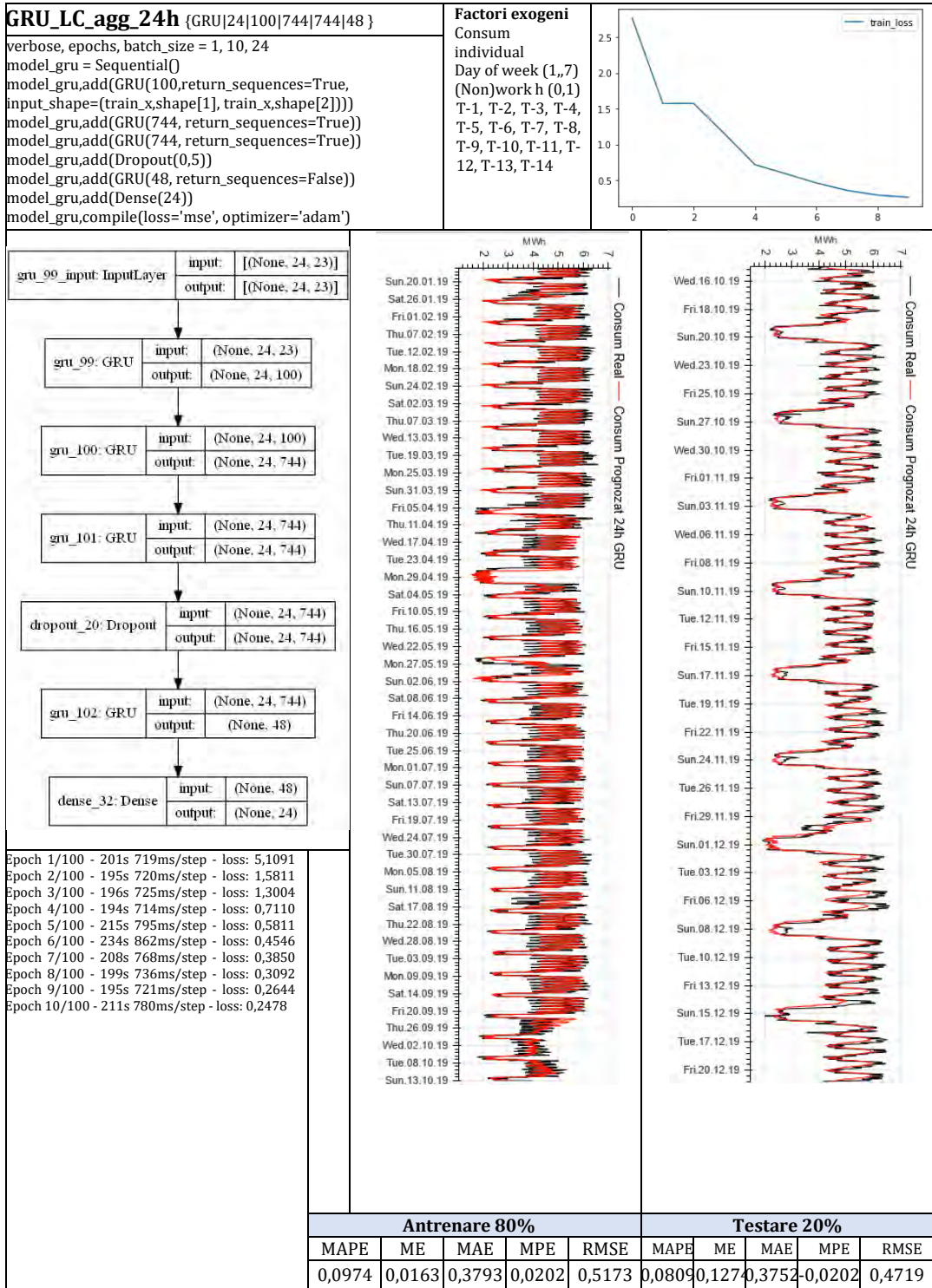


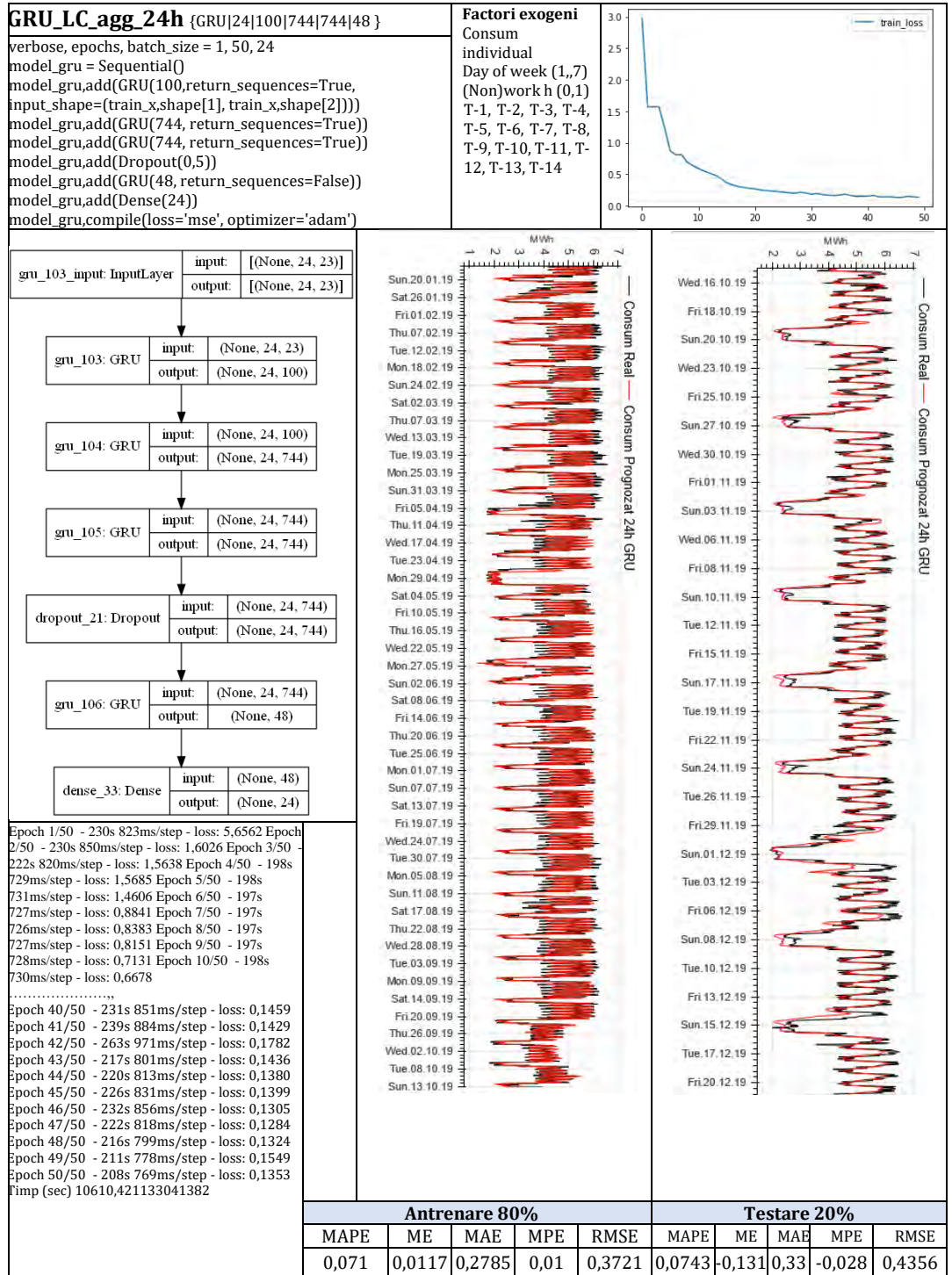
Epoch 1/100 - 201s 719ms/step - loss: 5.1091 Epoch 2/100 - 195s 720ms/step - loss: 1.5811 Epoch 3/100 - 196s 725ms/step - loss: 1.3004 Epoch 4/100 - 194s 714ms/step - loss: 0.7110 Epoch 5/100 - 215s 795ms/step - loss: 0.5811 Epoch 6/100 - 234s 862ms/step - loss: 0.4546 Epoch 7/100 - 208s 768ms/step - loss: 0.3850 Epoch 8/100 - 199s 736ms/step - loss: 0.3092 Epoch 9/100 - 195s 721ms/step - loss: 0.2644 Epoch 10/100 - 211s 780ms/step - loss: 0.2478 Epoch 11/100 - 203s 749ms/step - loss: 0.2293 Epoch 12/100 - 197s 727ms/step - loss: 0.2033 Epoch 13/100 - 195s 720ms/step - loss: 0.2044 Epoch 14/100 - 193s 711ms/step - loss: 0.2040 Epoch 15/100 - 192s 709ms/step - loss: 0.1829 Epoch 16/100 - 192s 708ms/step - loss: 0.2005

Epoch 90/100 - 192s 709ms/step - loss: 0,0532
Epoch 91/100 - 193s 712ms/step - loss: 0,0464
Epoch 92/100 - 192s 709ms/step - loss: 0,0854
Epoch 93/100 - 192s 708ms/step - loss: 0,0460
Epoch 94/100 - 192s 709ms/step - loss: 0,0341
Epoch 95/100 - 192s 707ms/step - loss: 0,0300
Epoch 96/100 - 192s 710ms/step - loss: 0,0286
Epoch 97/100 - 192s 707ms/step - loss: 0,0290
Epoch 98/100 - 192s 708ms/step - loss: 0,0269
Epoch 99/100 - 192s 708ms/step - loss: 0,0270
Epoch 100/100 - 192s 708ms/step - loss: 0,0264
Timp (sec) 19385,00232219696



Antrenare 80%					Testare 20%				
MAPE	ME	MAE	MPE	RMSE	MAPE	ME	MAE	MPE	RMSE
0,032	-0,023	0,128	-0,002	0,167	0,065	-0,121	0,301	-0,022	0,407





LISTĂ DE PUBLICAȚII

1) **Ungureanu, Stefan**, Vasile Topa, and Andrei C. Cziker 2021. "Deep Learning for Short-Term Load Forecasting—Industrial Consumer Case Study" *Applied Sciences* 11, no. 21: 10126. <https://doi.org/10.3390/app112110126>

2) **Ungureanu, Stefan**, Vasile Topa, and Andrei C. Cziker 2021. "Analysis for Non-Residential Short-Term Load Forecasting Using Machine Learning and Statistical Methods with Financial Impact on the Power Market" *Energies* 14, no. 21: 6966. <https://doi.org/10.3390/en14216966>

3) Anca Miron, Andrei C. Cziker, **Stefan Ungureanu**; Fuzzy logic controller for regulating the indoor temperature; 2021 9th International Conference on Modern Power Systems (MPS), 2021, pp. 1-6; DOI: 10.1109/MPS52805.2021.9492595.

4) Oltean Andrei, **Stefan Ungureanu**, Anca Miron and Andrei C. Cziker; IoT power monitoring device using Wi-fi and Arduino; 9th International Conference on Modern Power Systems (MPS), 2021, pp. 1-6, DOI: 10.1109/MPS52805.2021.9492651.

5) **Ungureanu Stefan**, Topa Vasile, Cziker Andrei, Miron, Anca, Darab, Cosmin; Application of Electricity Management Strategies for Lower Balancing Costs; EPE 2020 - Proceedings of the 2020 11th International Conference and Exposition on Electrical And Power Engineering, art. no. 9305524, pp. 345-349. DOI: 10.1109/EPE50722.2020.9305524

6) Miron, Anca, Cziker C. Andrei, **Ungureanu Stefan**, Beleiu G. Horia; The impact of multiple small pv units on distribution networks Romanian case-study; (2020) EPE 2020 - Proceedings of the 2020 11th International Conference and Exposition on Electrical And Power Engineering, art. no. 9305552, pp. 339-344. DOI: 10.1109/EPE50722.2020.9305552

7) Darab, C., Antoniu, T., Beleiu, H.G., Pavel, S., Birou, I., Micu, D.D., **Ungureanu, S.**, Cirstea, S.D.; Hybrid load forecasting using gaussian process regression and novel residual prediction; (2020) Applied Sciences (Switzerland), 10 (13), art. no. 4588, . Cited 2 times. DOI: 10.3390/app10134588

8) **Ungureanu, S.**, Topa, V., Cziker, A.; Industrial load forecasting using machine learning in the context of smart grid; (2019) 2019 54th International Universities Power Engineering Conference, UPEC 2019 - Proceedings, art. no. 8893540, . Cited 3 times. DOI: 10.1109/UPEC.2019.8893540

9) **Ungureanu Stefan**, Topa Vasile, Cziker Andrei.; Integrating the industrial consumer into smart grid by load curve forecasting using machine learning; (2019) Proceedings of 2019 8th International Conference on Modern Power Systems, MPS 2019, art. no. 8759707; DOI: 10.1109/MPS.2019.8759707