Univerzitet Singidunum

Departman za poslediplomske studije

Doktorska disertacija

UNAPREĐENJE MULTIVARIJANTNOG PREDVIĐANJA VREMENSKIH SERIJA PROIZVODNJE ELEKTRIČNE ENERGIJE IZ OBNOVLJIVIH IZVORA PRIMENOM MODELA REKURENTNIH NEURONSKIH MREŽA OPTIMIZOVANIH METAHEURISTIKAMA

Mentor:

Prof. dr Nebojša Bačanin-Džakula

Student:

Marijana Pavlov-Kagadejev

Broj indeksa: 480115/2022

Beograd, 2024.

Singidunum University

Graduate school

PhD Thesis

IMPROVEMENT OF MULTIVARIATE TIME SERIES PREDICTION OF ELECTRICAL ENERGY PRODUCTION FROM RENEWABLE SOURCES USING RECURRENT NEURAL NETWORK MODELS OPTIMIZED BY METAHEURISTICS

Mentor:

Professor Nebojša Bačanin-Džakula, PhD

Candidate:

Marijana Pavlov-Kagadejev

Index number: 480115/2022

Belgrade, 2024.

Mentor:

Prof. dr Nebojša Bačanin-Džakula, redovni profesor
Univerzitet Singidunum, Beograd

Članovi komisije:

Prof. dr Miodrag Živković, redovni profesor
Univerzitet Singidunum, Beograd
Predsednik komisije

Prof. dr Boško Nikolić, redovni profesor
Univerzitet u Beogradu, Elektrotehnički fakultet, Beograd
Član komisije

Posvećeno mom tati

ZAHVALNICA

Zahvaljujem se svima na pruženoj podršci i pomoći u toku izrade ove doktorske disertacije. Posebno se zahvaljujem svom mentoru prof. dr Nebojši Bačaninu-Džakuli, redovnom profesoru Univerziteta Singidunum u Beogradu na nesebičnoj pomoći i stručnim savetima, kao i članovima komisije prof. dr Miodragu Živkoviću, redovnom profesoru Univerziteta Singidunum u Beogradu i prof. dr Bošku Nikoliću, redovnom profesoru Univerziteta u Beogradu, Elektrotehničkog fakulteta u Beogradu. Zahvalnost dugujem i poslovodstvu Institututa za rudarstvo i metalurgiju Bor, koje mi je pomoglo u realizaciji ove disertacije. Zahvaljujem se svima koji su mi pružali podršku u teškim životnim trenucima.

Veliku zahvalnost dugujem i svojoj porodici, sinu Mihajlu i suprugu Vladimiru, na strpljenju, razumevanju i podršci. Najveću zahvalnost dugujem svom ocu Toši, koji je uvek bio moja najveća podrška, ali čija se želja da dočeka ovaj dan nije ispunila. Iz tog razloga je ova disertacija upravo njemu posvećena.

Unapređenje multivarijantnog predviđanja vremenskih serija proizvodnje električne energije iz obnovljivih izvora primenom modela rekurentnih neuronskih mreža optimizovanih metaheuristikama

Sažetak – Proizvodnja električne energije iz obnovljivih izvora predstavlja važan deo savremenih energetskih mreža. Da bi se izbegli gubici i održala ravnoteža između proizvodnje i potražnje, potrebne su stabilne metode za predviđanje proizvodnje električne energije iz obnovljivih izvora. U ovoj disertaciji predložen je pristup koji kombinuje tehnike dekompozicije signala sa modelom duge kratkoročne memorije (eng. Long Short-Term Memory - LSTM), koja je korišćena za predviđanje proizvodnje energije vetra. Parametri LSTM modela optimizovani su pomoću unapređenog metaheurističkog algoritma. LSTM mreže pokazuju izuzetne performanse kod rešavanja problema predviđanja vremenskih serija, a dalje poboljšanje performanisi može se postići optimizacijom hiperparametara pomoću unapređene verzije algoritma pretrage reptila (eng. Reptile Search Algorithm - RSA), nazvanog unapređeni algoritam pretrage reptila (eng. Enhanced Reptile Search Algorithm - ERSA). ERSA je na početku evoluiran na standardnim CEC2019 benčmark funkcijama, a kasnije je primenjen na konkretne probleme. Optimizovani LSTM model je testiran na dva dataseta o proizvodnji vetra. Predviđanje je izvršeno za slučaj bez dekompozicije signala i primenom tehnika varijacione (eng. Variational Mode Decomposition - VMD) i empirijske (eng. Empirical Mode Decomposition - EMD) dekompozicije, za jedan, dva i tri koraka unapred. Izvršeno je poređenje rezultata dobijenih simulacijom, sa LSTM mrežama koje su optimizovane pomoću drugih savremenih metaheuristika. Može se primetiti da predložena metodologija prevazilazi svoje konkurente, što je potvrđeno statističkom analizom. Na kraju ove disertacije izvršena je i Shapley analiza (eng. SHapley Additive Explanations - SHAP) modela sa najboljim performansama, gde je dat uticaj svake karakteristike na predviđanje modela.

Ključne reči: energija vetra, LSTM, metaheurističke optimizacije, RSA, SHAP analiza.

Improvement of Multivariate Time Series Prediction of Electrical Energy Production from Renewable Sources Using Recurrent Neural Network Models Optimized by Metaheuristics

Abstract – Electricity production from renewable sources represents an important part of modern energy grids. To minimize losses and maintain a balance between production and demand, stable methods for predicting electricity production from renewable sources are necessary. This dissertation proposes an approach that combines signal decomposition techniques with Long Short-Term Memory (LSTM) neural networks used for wind energy production prediction. The parameters of the LSTM model are optimized using an enhanced metaheuristic algorithm. LSTM networks exhibit exceptional performance in solving time series prediction problems, and further performance improvement can be achieved by optimizing hyperparameters using an enhanced version of the Reptile Search Algorithm (RSA), named Enhanced Reptile Search Algorithm (ERSA). ERSA is initially evaluated on standard CEC2019 benchmark functions and later applied to specific problems. The optimized LSTM model is tested on two datasets of wind production. Prediction is performed for the case without signal decomposition and using Variational Mode Decomposition (VMD) and Empirical Mode Decomposition (EMD) techniques, for one, two, and three steps ahead. A comparison of simulation results is made with LSTM networks optimized using other contemporary metaheuristics. It can be observed that the proposed methodology outperforms its competitors, which is confirmed by statistical analysis. Finally, Shapley Additive Explanations (SHAP) of the best-performing model is conducted, providing insight into the impact of each feature on the model's prediction.

Keywords: wind power generation, long short-term memory networks, metaheuristics optimization, reptile search algorithm, SHapley additive explanations.

Spisak slika

1 Konfiguracija HAWT rotora [81] 34
2 Tipična kriva snage vetrogeneratora 35
3 Cirkulacija površinskih vetrova [81] 36
4 Ilustracija geostrofskog vetra 37
5 Tipičan dijagram promene brzine vetra sa vremenom u kratkom periodu 41
6 Diskretni udari vetra: (a) amplituda; (b) vreme uzdizanja; (c) maksimalna varijacija naleta vetra; (d) vreme opadanja. 41
7 Klase kompleksnosti problema odlučivanja 51
8 Podela modela mašinskog učenja 55
9 a) Nadgledano i b) Nenadgledano učenje [11] 55
10 Blok-šema procesa nadgledanog učenja 56
11 Biološki neuron 60
12 Matematički model neurona [82] 61
13 Primer neuronske mreže [37] 63
14 Neuronska mreža sa n + 1 ulazom, slojem od Nc skrivenih neurona sa sigmoidnom aktivacionom funkcijom, i linearnim izlaznim neuronom [37] 64
15 Poređenje performansi ML i DL [106] 65
16 Model rekurentne neuronske mreže [46] 65
17 LSTM ćelija 67
18 Arhitektura BiLSTM mreže 68
19 GRU ćelija 70
20 Poređenje GS i RS algoritama za optimizaciju hiperparametara [52] . 75
21 Grafička reprezentacija integracije funkcije f(x) metodom Monte Karlo[73] 83
22 Lokalna pretraga 84
23 Generacija u EA 88
24 Genotip i fenotip u EA 88
25 Ilustracija ukrštanja kod binarnog GA 90
26 Ilustracija izbegavanja lokalnog minimuma kod SA 95
27 Način na koji mravi pronalaze optimalni put od gnezda do hrane . . . 98
28 Ilustracija načina kretanja čestica u svakoj iteraciji kod PSO [119] . . 100
29 Način kretanja čestice i ažuriranje brzine 101

30 Ponašanje kolonije pčela nakon otkrivanja izvora hrane [119] 105
31 Opkoljavanje plena, kada je (t $\leq \frac{T}{2}$)[3] 114
32 Napad na plen, kada je ($t > \frac{T}{2}$) [3] 116
33 Grafovi brzine konvergencije za neke proizvoljne CEC2019 funkcije . . 129
34 Wind dataset podeljen na podatke za trening, validaciju i testiranje . 131
35 Wind farm dataset podeljen na podatke za trening, validaciju i testiranje 132
36 VMD modovi i reziduali ulaznih karakteristika za Wind dataset . . . 133
37 VMD modovi i reziduali ulaznih karakteristika za Wind farm dataset 134
38 EMD IMF-ovi i reziduali ulaznih karakteristika za Wind dataset . . 135
39 EMD IMF-ovi i reziduali ulaznih karakteristika za Wind farm dataset 136
40 Diagram toka procesa predviđanja 137
41 Wind dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku bez dekompozicije 140
42 Wind dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku bez dekompozicije 142
43 Wind dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku bez dekompozicije 142
44 Wind dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku sa primenom VMD dekompozicije 145
45 Wind dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku sa primenom VMD dekompozicije 146
46 Wind dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku sa primenom VMD dekompozicije 146
47 Wind dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku sa primenom EMD dekompozicije 147
48 Wind dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku sa primenom EMD dekompozicije 149
49 Wind dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku sa primenom EMD dekompozicije 149
50 Wind farm dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku bez dekompozicije 153

51 Wind farm dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku bez dekompozicije 153
52 Wind farm dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku bez dekompozicije 154
53 Wind farm dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku sa primenom VMD dekompozicije 155
54 Wind farm dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku sa primenom VMD dekompozicije . . . 157
55 Wind farm dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku sa primenom VMD dekompozicije 157
56 Wind farm dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku sa primenom EMD dekompozicije 160
57 Wind farm dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku sa primenom EMD dekompozicije . . . 161
58 Wind farm dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku sa primenom EMD dekompozicije 161
59 Uticaj karakteristika na najbolji LSTM model za Wind dataset bez dekompozicije, određen SHAP analizom 166
60 Uticaj karakteristika na najbolji LSTM model za Wind dataset sa VMD dekompozicijom, određen SHAP analizom 166
61 Uticaj karakteristika na najbolji LSTM model za Wind farm dataset bez dekompozicije, određen SHAP analizom 167
62 Uticaj karakteristika na najbolji LSTM model za Wind farm dataset sa VMD dekompozicijom, određen SHAP analizom 168

Spisak tabela

1 Analogija između prirodne i veštačke kolonije pčela 106
2 Pregled CEC2019 benčmark funkcija. 127
3 Rezultati RSA dobijeni korišćenjem CEC2019 test funkcija. 128
4 Hiperparametri LSTM modela sa odgovarajućim ograničenjima 138
5 Wind dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, bez dekompozicije 139
6 Wind dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje bez dekompozicije 141
7 Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra bez dekompozicije 142
8 Wind dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, sa primenom VMD dekompozicije 143
9 Wind dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje sa primenom VMD dekompozicije 144
10 Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra sa primenom VMD dekompozicije 145
11 Wind dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, sa primenom EMD dekompozicije 146
12 Wind dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje sa primenom EMD dekompozicije 148
13 Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra sa primenom EMD dekompozicije 149
14 Poređenje najboljeg modela sa savremenim modelima za predikciju koji su primenjeni na Wind dataset 150
15 Wind farm dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, bez dekompozicije 151
16 Wind farm dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje bez dekompozicije 152
17 Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra bez dekompozicije 153

18 Wind farm dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, sa primenom VMD dekompozicije 154
19 Wind farm dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje sa primenom VMD dekompozicije 156
20 Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra sa primenom VMD dekompozicije 157
21 Wind farm dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, sa primenom EMD dekompozicije 158
22 Wind farm dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje sa primenom EMD dekompozicije 159
23 Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra sa primenom EMD dekompozicije 160
24 Poređenje najboljeg modela sa savremenim modelima za predikciju koji su primenjeni na Wind farm dataset 162
25 Rezultati Shapiro-Wilk testa za analizu pojedinačnih problema 163
26 Rezultati Shapiro-Wilk testa za analizu višestrukih problema 163
27 Rezultati Friedman testa 164
28 Rezultati Friedman aligned testa 164
29 Holm's step-down procedura 165

Algoritmi

1 LS algoritam 84
2 Pseudo-kod jednostavnog GA algoritma 92
3 ACO algoritam 98
4 PSO algoritam 102
5 ABC algoritam 106
6 Pseudo-kod FA 109
7 Pseudo-kod RSA 117
8 Predložena procedura inicijalizacije koja uključuje QRL metod 118
9 ERSA Pseudo-kod 121

Sadržaj

Spisak slika **6**

Spisak tabela **9**

1 Uvod **16**

1.1 Pregled dosadašnjih istraživanja i literature 19

1.2 Cilj, hipoteze i metode istraživanja 21

1.2.1 Cilj istraživanja 21

1.2.2 Hipoteze 21

1.2.3 Metode primenjene u istraživanju 22

1.3 Naučni doprinos 23

1.4 Sadržaj doktorske disertacije 24

2 Koncept proizvodnje energije iz obnovljivih izvora **26**

2.1 Vrste obnovljivih izvora energije 27

2.1.1 Solarna Energija 27

2.1.2 Energija Vetra 27

2.1.3 Biomasa 28

2.1.4 Geotermalna energija 30

2.1.5 Hidroenergija 31

2.1.6 Energija okeana 32

2.2 Proizvodnja električne energije iz energije vetra 32

2.2.1 Dizajn modernih vetrogeneratora 33

2.2.2 Predviđanje izlazne snage 34

2.2.3 Karakteristike i resursi vetra 35

2.2.4 Globalno poreklo vetra 35

2.2.5 Mehanika kretanja vetra 36

2.2.6 Ostali oblici atmosferske cirkulacije 38

2.3 Prostorne i vremenske karakteristike vetra 39

2.3.1 Varijacija u vremenu 39

2.3.2 Varijacije u zavisnosti od lokacije i smera vetra 42

3 Optimizacioni problemi **43**

3.1 Podela optimizacionih problema 45

3.1.1 Podela optimizacionih problema prema tipu varijabli 45

3.1.2 Podela optimizacionih problema u zavisnosti od ograničenja . 47

3.1.3 Podela optimizacionih problema na osnovu broja funkcija cilja 48

3.2 Klase kompleksnosti 50

4 Metode i algoritmi mašinskog učenja **53**

4.1 Vrste podataka 54

4.2 Pregled modela mašinskog učenja 54

4.2.1 Nadgledano učenje 56

4.2.2 Nenadgledano učenje 57

4.2.3 Polunadgledano učenje 58

4.2.4 Pojačano učenje 59

4.3 Neuronske mreže 59

4.3.1 Biološki i veštački neuron 59

4.3.2 Vrste neuronskih mreža 62

4.4 Duboko učenje 64

4.5 RNN 65

4.5.1 LSTM 66

4.5.2 BiLSTM 68

4.5.3 GRU 69

4.6 Optimizacija hiperparametara 70

4.6.1 Metodi za optimizaciju hiperparametara 73

5 Taksonomija optimizacionih algoritama **77**

5.1 Kategorizacija metaheurističkih algoritama 78

5.2 Metaheuristike koje nisu inspirisane prirodom 81

5.2.1 Monte Karlo metod 81

5.2.2 Lokalna pretraga 83

5.2.3 Gramzivi algoritmi 85

5.2.4 Tabu pretraga 85

5.3 Metaheuristike inspirisane prirodom 86

5.3.1 Evolutivni algoritmi 87
5.3.2 Genetski algoritmi 88
5.3.3 Simulirano kaljenje 92

6 Metaheuristike inspirisane inteligencijom rojeva 97
6.1 Optimizacija mravljom kolonijom 97
6.2 Optimizacija rojem čestica 99
6.3 Optimizacija veštačkom kolonijom pčela 103
6.4 Optimizacija inspirisana ponašanjem svitaca 108

7 Prikaz implementacije hibridnih metoda metaheuristika i modela rekurentnih neuronskih mreža 112
7.1 Originalni algoritam pretrage reptila (RSA) 112
7.1.1 Ponašanje krokodila u prirodi 112
7.1.2 Faza inicijalizacije 114
7.1.3 Faza opkoljavanja (exploracija) 114
7.1.4 Faza lova (eksploatacija) 116
7.2 Predloženi modifikovani pristup RSA algoritmu 117
7.2.1 Nova šema inicijalizacije 117
7.2.2 Procedura za očuvanje raznovrsnosti populacije 118
7.2.3 Struktura predloženog algoritma 120
7.3 Dekompozicija signala 120
7.3.1 Varijaciona dekompozicija 121
7.3.2 Empirijska dekompozicija 122
7.4 Metrike performansi generisanih modela 124
7.5 SHAP analiza 124
7.6 Eksperimenti sa standardnim funkcijama sa ograničenjima opsega . . 125
7.7 Pregled korišćenih skupova podataka 129
7.7.1 Wind dataset 130
7.7.2 Wind farm dataset 130
7.8 Primena tehnika dekompozicije 132
7.9 Postavka eksperimenta 137

8 Postignuti rezultati, uporedna analiza i diskusija 139

8.1 Eksperimentalni rezultati za Wind dataset 139

8.1.1 Wind dataset sa VMD dekompozicijom 140

8.1.2 Wind dataset sa EMD dekompozicijom 145

8.1.3 Poređenje sa drugim modelima za Wind dataset 150

8.2 Eksperimentalni rezultati za Wind farm dataset 150

8.2.1 Wind farm dataset sa VMD dekompozicijom 151

8.2.2 Wind farm dataset sa EMD dekompozicijom 155

8.2.3 Poređenje sa drugim modelima za Wind farm dataset 162

8.3 Validacija - statistički testovi 162

8.4 Interpretacija najboljeg modela i SHAP analiza 165

8.4.1 Wind dataset SHAP analiza 166

8.4.2 Wind farm dataset SHAP analiza 167

9 Zaključak 169

10 Literatura 170

1 Uvod

Proizvodnja električne energije korišćenjem obnovljivih izvora je ključna iz više razloga. Generisanje energije korišćenjem vetra i solarnih panela po prirodi je promenljivo [76], što znači da njihov izlaz može oscilirati u zavisnosti od vremenskih uslova i drugih faktora. Precizni podaci o proizvodnji energije omogućavaju mrežnim operaterima i energetskim kompanijama da bolje predvide ove fluktuacije i uspešnije upravljaju ukupnim snabdevanjem energijom, kako bi osigurali stabilnu i pouzdanu mrežu [29]. Tačni podaci o proizvodnji električne energije su neophodni i u svrhu fakturisanja, jer osiguravaju da energetski provajderi dobiju adekvatnu nadoknadu za proizvedenu energiju. Konačno, precizni podaci o proizvodnji energije su važni, jer pomažu prilikom donošenja odluka o investicijama u energetsku infrastrukturu, uticaju na životnu sredinu i razvoju tehnologije obnovljive energije [10]. Ukratko, tačni podaci o proizvodnji energije su ključni kako bio se osigurala efikasnost, pouzdanost i održivost energetskih sistema.

Istorijski podaci o proizvodnji energije mogli bi biti korisni indikatori za buduće prognoze [112]. Međutim, nekoliko međusobno povezanih parametara utiče na proizvodnju energije. Svi ovi parametri su podložni promenama, što čini predviđanje veoma teškim zbog kompleksnosti signala. Tehnike dekompozicije poput VMD[1] (eng. Variational Mode Decomposition) [102] i EMD (eng. Empirical Mode Decomposition) [103] imaju potencijal da se nose sa kompleksnim signalima, tako što razlažu složeni signala na jednostavnije komponente, koje je moguće lakše analizirati. Obe razlažu signal tako što formiraju set varijacionih modova, ili inherentnih modnih funkcija (eng. Intrinsic Mode Functions - IMFs), redom, koji obuhvataju osnovne frekvencijske komponente signala. Svaka IMF, kao i svaki varijacioni mod, predstavlja različitu frekvencijsku komponentu signala, pri čemu najviši frekvencijski modovi obuhvataju najbrže i tranzitne promene, a najniži frekvencijski modovi obuhvataju sporije, stabilnije promene. Analizom IMF-a ili varijacionih modova zasebno, može se dobiti uvid u osnovne obrasce i dinamiku signala koja može biti prikrivena njegovom kompleksnošću. Uopšteno, tehnike dekompozicije poput VMD i EMD pružaju moćan pristup za rešavanje kompleksnih signala, omogućavajući istraživačima dublje

[1]Veliki broj termina iz računarske tehnike nema ustaljen i opšte prihvaćen naziv na srpskom jeziku. Iz tog razloga će nakon datog naziva na srpskom jeziku u zagradi biti dat pun naziv i akronim na engleskom jeziku, koji će se koristiti u daljem tekstu disertacije.

razumevanje njihove osnovne dinamike.

Nove tehnike veštačke inteligencije imaju potencijal da tačno prognoziraju proizvodnju energije dobijene iz vetra na nekoliko načina. Modeli se mogu trenirati na velikim skupovima istorijskih podataka o vremenu i o vetru, kako bi bolje predvideli brzine i pravac vetra, koji su ključni faktori u proizvodnji energije iz vetra. Ovo se može iskoristiti kako bi se poboljšala tačnost predviđanja proizvodnje energije i pomoglo energetskim provajderima da bolje sagledaju fluktuacije u proizvodnji energije vetra. LSTM (eng. Long Short-Term Memory) [55] je vrsta rekurentnih neuronskih mreža dizajnirana da modeluje sekvence i vremenske serije. LSTM mreže su pogodne za vremenske serije jer mogu pronaći i zapamtiti dugoročne zavisnosti i obrasce koji se javljaju u podacima tokom vremena, izbegavajući pri tome problem nestajanja gradijenta (eng. gradijent vanishing) koji može nastati u tradicionalnim rekurentnim neuronskim mrežama. Ove mreže koriste sistem *kapija* (eng. gate) za modeliranje prenosa podataka i kontrolu memorije mreže, što omogućava selektivno zaboravljanje ili pamćenje informacija iz prethodnih vremenskih koraka. Ovo čini LSTM mreže veoma efikasnim za modeliranje vremenskih serija sa kompleksnom vremenskom dinamikom. Ovaj tip mreža pruža snažan alat za rad sa vremenskim serijama, omogućavajući tačne prognoze i uvid u suptilne obrasce i veze među podacima. Međutim, kao i mnogi algoritmi, oni sadrže skup hiperparametara koji zahtevaju odgovarajuću optimizaciju kako bi se postigli željeni rezultati.

Hiperparametri su bitni za modele veštačke inteligencije, jer mogu značajno uticati na performanse modela [98]. Ove parametre model ne uči tokom treninga, već ih korisnik definiše pre treninga modela. Izbor adekvatnih hiperparametara za model je ključan, jer odabir pogrešnih vrednosti može dovesti do loših performansi, sporog konvergiranja ili preprilagođavanja (eng. overfitting) modela. Optimizacija hiperparametara uključuje odabir optimalnih vrednosti hiperparametara za postizanje najboljih mogućih rezultata za dati model. To se obično postiže kombinacijom pokušaja i grešaka i automatizovanih tehnika. Proces uključuje trening modela sa različitim vrednostima hiperparametara i evaluaciju performansi na validacionom skupu. Zatim se biraju vrednosti hiperparametara za koje se dobija optimizovan model.

Metaheuristički algoritmi predstavljaju moćnu klasu optimizacionih algoritama

koji se mogu primeniti na optimizaciju hiperparametara u modelima veštačke inteligencije koji se koriste za predviđanje [15]. Iterativna optimizacija je dizajnirana da pretražuje i analizira prostor pretrage mogućih rešenja, ne praveći pretpostavke o funkciji cilja ili strukturi problema. Pogodni su za optimizaciju hiperparametara, jer se mogu primeniti na visokodimenzionalnim i nekonveksnim prostorima istraživanja sa mnogo lokalnih optimuma. Metaheuristički algoritmi nude obećavajući pristup optimizaciji hiperparametara u modelima veštačke inteligencije koji se koriste za predviđanje i mogu pomoći u prevazilaženju nekih ograničenja, koja postoje kod tradicionalnih metoda optimizacije. Koristeći snagu iterativne pretrage i istraživanja, metaheuristički algoritmi mogu pomoći u određivanju optimalnih vrednosti hiperparametara i poboljšanju tačnosti i performansi modela. Metaheuristički algoritmi roja predstavljaju grupu optimizacionih algoritama, koja je inspirisana kolektivnim socijalnim ponašanjem insekata i životinja. Koncept decentralizovane samoorganizacije je ključan kod ovih algoritama, gde grupa jedinki međusobno deluje kako bi postigla zajednički cilj. Prateći jednostavna pravila, kompleksna ponašanja se javljaju na globalnom nivou. Ovi algoritmi imaju sposobnost rešavanja NP-teških (eng. non-deterministic polynomial time) optimizacionih problema, uz razumnu potrošnju vremena i resursa, što često predstavlja problem kod tradicionalnih metoda. Međutim, teorema o nepostojanju besplatnog ručka (eng. No Free Lunch -NFL) [127], govori o tome da nijedna pojedinačna metodologija nije najbolja za sve probleme, pa se preferira individualni pristup. Iz tog razloga je potrebno sprovesti temeljna istraživanja kako bi se dodatno unapredile tehnike i metode.

U ovoj disertaciji korišćen je metahrustički algoritam sa veoma dobrim performansama, algoritam pretrage reptila (eng. Reptile Search Algorithm -RSA) [2]. Ovo meta-učenje brzo prilagođava model novim zadacima. To radi tako što model trenira na skupu zadataka tokom fiksnog broja iteracija, a zatim ga fino prilagođava novim zadacima koristeći samo nekoliko ažuriranja gradijenta. RSA ima za cilj da nauči optimalnu inicijalizaciju parametara modela, omogućavajući im da se brzo prilagode novim zadacima sa minimalnim izmenama.

Motivacija za sprovedeno istraživanje bila je dalje istraživanje i proširenje razumevanja novog RSA i njegove potencijalne upotrebe za podešavanje hiperparametara. Dodatno, ključni motiv bio je utvrditi da li se ovaj metaheuristički algoritam, koji

već ima izuzetne performanse, može dodatno poboljšati hibridizacijom s drugim poznatim snažnim optimizatorima. Na kraju, ovo istraživanje ima za cilj uvođenje robustne metode zasnovane na veštačkoj inteligenciji, koja je unapređena uvođenjem ovog metaheurističkog algoritma, kako bi se bolje suočila sa realnim problemom predviđanja proizvodnje el. energije iz energije vetra.

U ovoj disertaciji dat je predlog nove metode za prognoziranje proizvodnje energije vetra na osnovu vremenskih serija meteoroloških i istorijskih faktora. Kako bi se uzele u obzir kompleksnosti uzrokovane promenljivom prirodom ovih prediktora, koriste se dve tehnike dekompozicije signala, VMD i EMD. Obrađeni podaci se formulišu kao vremenske serije, pri čemu se kao ulazni podaci koristi šest prethodnih vrednosti, kako bi se obučile LSTM neuronske mreže za kreiranje prognoze tri koraka unapred. Da bi se izvršila optimizacije performansi modela, primenjuju se metahrustički algoritmi za selekciju hiperparametara. Nekoliko metahrustičkih algoritama se evaluira, a uvodi se i nova modifikovana verzija RSA nazvana ERSA (eng. Enhanced Reptile Search Algorithm). Performanse algoritma obično se prvo evaluiraju na skupu standardizovanih funkcija za testiranje. Iz tog razloga, modifikovani metahrustički algoritam je evaluiran na širokom spektru standardnih CEC2019 funkcija za testiranje sa ograničenjima pre nego što je primenjen na stvarni problem. Nakon ovih evaluacija, svaki LSMT sa dekompozicijom optimizovan metaheuristikom evaluiran je na dva skupa podataka iz stvarnog sveta koji potiču od dva vetroparka u različitim delovima sveta kako bi se odredile njihove performanse. Modeli sa najboljim performansama su interpretirani korišćenjem SHAP metoda (eng. SHapley Additive exPlanations) [79], kako bi se odredili faktori koji imaju najveći uticaj na predviđanja modela.

1.1 Pregled dosadašnjih istraživanja i literature

Nekoliko studija je pokazalo efikasnost LSTM modela u tačnom predviđanju proizvodnje energije vetra na kratkoročnim i dugoročnim horizontima. Jedna takva studija [109] predložila je model zasnovan na LSTM za kratkoročno prognoziranje energije vetra koji je uključivao i meteorološke i podatke o energiji. Autori su pokazali da je dati model nadmašio tradicionalne modele vremenskih serija i druge algoritme mašinskog učenja, postižući visoku tačnost i robustnost na različitim lokacijama i pri različitim vremenskim uslovima.

Drugo istraživanje [78] fokusiralo se na dugoročno predviđanje energije vetra upotrebom LSTM mreže. Autori su predstavili hibridni model koji je kombinovao LSTM, wavelet transformaciju i analizu glavnih komponenti kako bi obuhvatili i vremenske i prostorne varijacije u podacima o energiji vetra. Predloženi model dao je bolje rezultate od tradicionalnih statističkih modela i drugih algoritama mašinskog učenja, sa poboljšanom tačnošću i robustnošću za prognozu za duži vremenski period.

Pored LSTM neuronskih mreža, istraživane su i druge tehnike kako bi se poboljšala preciznost prognoziranja proizvodnje energije vetra, uključujući upotrebu tehnika dekompozicije poput VMD i EMD. Istraživači su u radu [135] predložili pristup zasnovan na VMD tehnici. Kako bi se poboljšala tačnost predviđanja proizvodnje energije vetra, primenjena je dekomponzicija vremenske serije na različite frekvencijske komponente, a zatim je primjen zaseban model na svaku komponentu. Autori su pokazali da pristup utemeljen na VMD tehnici nadmašuje tradicionalne modele vremenskih serija, kao i druge algoritme mašinskog učenja, postižući visoku tačnost i robusnost na različitim lokacijama i vremenskim uslovima.

Optimizacija hiperparametara ključan je aspekt mašinskog učenja i ispitivana je u kontekstu predviđanja proizvodnje energije vetra korištenjem metaheurističkih algoritama. Prethodni radovi [110] predložili su pristup zasnovan na algoritmu vatrometa (eng. Fireworks Algorithm) za optimizaciju hiperparametara LSTM neuronskih mreža za prognozu energije vetra. Autori su pokazali da je predloženi pristup postigao veću tačnost prognoze u poređenju sa drugim tehnikama optimizacije, što ukazuje na važnost optimizacije hiperparametara za preciznu prognozu proizvodnje energije vetra. Osim toga, metaheuristike su primenjene u optimizaciji u nekoliko područja i pokazale su značajne rezultate, uključujući prognoziranje cena sirove nafte [65] i nauke o životnoj sredini [66].

Drugi pristup za povećanje kompatibilnosti modela za prognoziranje proizvodnje električne energije vetra je korišćenje SHAP [80] vrednosti. Ove vrednosti pružaju način za tumačenje odluka modela mašinskog učenja, određujući doprinose koje pružaju dostupne karakteristike na konačan rezultat. Korišćenje SHAP vrednosti može pomoći u razumevanju faktora koji utiču na tačnost predviđanja energije vetra koristeći modele zasnovane na LSTM. SHAP vrednosti pružaju sveobuhvatnije i intuitivnije razumevanje performansi modela, u poređenju sa tradicionalnim metrikama

evaluacije.

1.2 Cilj, hipoteze i metode istraživanja

1.2.1 Cilj istraživanja

Obnovljiva energija igra ključnu ulogu u rešavanju hitnih ekoloških izazova naše planete [5]. Korišćenjem izvora kao što su solarna i energija vetra može se značajno smanjiti emisija gasova sa efektom staklene bašte, a samim tim i klimatske promene [101]. Prihvatanje obnovljive energije ne samo da štiti okolinu, već takođe promoviše energetsku sigurnost, nova radna mesta i zdraviju budućnost za generacije koje dolaze.

Međutim, metode za generisanje energije na obnovljiv način još uvek se razvijaju. Da bi se olakšalo masovno usvajanje, potrebno je prevazići mnoge izazove [143]. Jedan od glavnih izazova dolazi u obliku povećane pouzdanosti. Mogućnost predviđanja proizvodnje energije energetskih sistema, može poboljšati pouzdanost na duge staze, povećavajući održivost obnovljivih sistema.

Potencijal metaheurističkih algoritama za optimizaciju hiperparametara je dobro razvijen pristup [120], međutim, još uvek nije dovoljno istražen za primenu LSTM mreža za predviđanje generisanja energije vetra. Nove i moćnije tehnike se neprestano razvijaju [83, 20]. Potencijal nedavno razvijenog RSA [2] još uvek nije istražen i implementiran za predviđanje proizvodnje energije. Kao relativno nov pristup, postoji veliki potencijal poboljšanja ovog algoritma hibridizacijom.

Tehnike dekompozicije nude još jednu tehniku koja može poboljšati obuku modela, a sa razvojem novih metoda potrebno je eksperimentisati kako bi se utvrdio njihov potencijal za rešavanje ovog sve hitnijeg problema. Tehnike interpretacije modela [39] postaju sve važnije za izgradnju pouzdanijih modela i stabilnijih sistema. Ova disertacija ima za cilj da poveća broj dostupnih tehnika koje će se primenjivati za predviđanje proizvodnje energije iz obnovljivih izvora.

1.2.2 Hipoteze

Opšta hipoteza, od koje se krenulo u istraživanju za potrebe izrade doktorske disertacije može da se formuliše kao:

Hipoteza X: Primenom do sada nedovoljno korišćenih modela rekurnetnih neuronskih mreža za multivarijantno predviđanje vremenskih serija proizvodnje energije iz obnovljivih izvora, moguće je generisati bolja rešenja u smislu brzine konvergincije i kvaliteta (performansi, MSE, R^2), čime će se unaprediti rezultati koje su druge metode optimizacije postigle za istu instancu problema.

Posebna hipoteza koja proizilazi iz opšte i koja se odnosi na obrađivanje delova predmeta istraživanja glasi:

X0.1. Moguće je adaptirati i optimizovati postojeće implementacije modela rekurentnih neuronskih mreža (eng. Recurent Neural Network - RNN) za mutlivarijantno predviđanje vremenskih serija proizvodnje el. energije iz obnovljivih izvora. Daljim preciziranjem navedenih posebnih hipoteza, formulišu se pojedinačne koje se odnose na elementarne činioce predmeta istraživanja:

- X0.1.1. Moguće je adaptirati postojeće implementacije metaheuristika koje do sada nisu primenjivane u ovom domenu, za optimizaciju RNN za predviđanje vremenskih serija.
- X0.1.2. Moguće je unaprediti postojeće implementacije metaheuristika rojeva, koje su ranije već bile primenjene u ovom domenu, za optimizaciju RNN za predviđanje vremenskih serija.

1.2.3 Metode primenjene u istraživanju

Za razvoj hipoteza koje su usklađene sa predmetom i ciljem istraživanja, korišćene su različite istraživačke metode koje su široko prihvaćene u praksi i potvrđene u naučnoj zajednici. Ove metode uključuju analitičko-sintetičku metodu, deduktivno-induktivni pristup, komparativno-kvantitativnu analizu, empirijske tehnike, statističku analizu, modeliranje i eksperimentalni pristup za potvrđivanje hipoteza.

Kako bi se izvršila analiza sadržaja relevantne stručne i naučne literature u ovoj oblasti, kao i analiza činjenica koje su publikovane putem elektronskih i drugih medija, primenjena je *analitička metoda*. Da bi se došlo do pouzdanih zaključaka, *metodom sinteze* izvršeno je grupisanje svih podataka koji su prikupljeni. U cilju dolaska do adekvatnih zaključaka, istraživanje je usmereno od opšteg ka pojedinačnom i od pojedinačnog ka opštem primenom *deduktivno-induktivnog metoda*. U okviru *komparativno-kvantitativne analize*, koja je u skladu sa temom kojom se

istraživanje bavi i ciljevima istraživanja, metoda analize izabrana je kao najpogodniji pristup. *Metoda kvantitativne analize* korišćena je prilikom analize konkretnih prikupljenih rezultata primene tehnika, metoda i metaheuristika za rešavanje problema multivarijantnog predviđanja vremenskih serija proizvodnje energije iz obnovljivih izvora. Poređenje dobijenih rezultata optimizacije i izvođenje zaključka o stvarnom unapređenju rešavanja ovog problema primenom algoritma inteligencije rojeva, izvršeno je *komparativnom analizom*. Primena *empirijske metode* omogućila je da se prikupljeni primarni podaci razmotre u svetlu postojećih teorijskih i empirijskih nalaza, kao i sekundarnih izvora podataka, kako bi se početne istraživačke hipoteze testirale. Praktičnim eksperimentima (pokretanjem algoritama) došlo se do primarnih podataka. *Statističkom analizom* obuhvaćeno je korišćenje statističkih metoda i podataka prilikom obrade dobijenih rezultata, u skladu sa ciljevima istraživanja. *Statističko-ekonometrijske metode* trenda i korelacije korišćene su za određivanje stepena i pravca međusobne zavisnosti rezultata različitih metaheuristika, metoda i tehnika. *Modeliranje* podrazumeva izradu matematičkih modela koji validno predstavljaju problem koji se proučava i rešava. Za rešavanje optimizacionih problema posmatraju se matematički modeli i formulacije, kao i funkcije ciljeva problema istraživanja. Takođe se, u eksperimentalnom delu implementacije, sagledavaju tehnike metaheuristika u rešavanju problema.

1.3 Naučni doprinos

Naučni doprinosi ove doktorske disertacije su:

- Predlog modifikovane varijante RSA, koji poboljšava dobre performanse originala.
- Uvođenje metodologije optimizovane metaheuristikom uz pomoć dekompozicije za predviđanje vremenskih serija proizvodnje el. energije iz energije vetra.
- Interpretacija modela sa najboljim performansama korišćenjem SHAP analize, radi boljeg razumevanja faktora koji najviše doprinose proizvodnji el. energije iz energije vetra.

Rezultati ove doktorske disertacije objavljeni su u renomiranom svetskom časopisu *Artificial Intelligence Review*, koji je kategorizovan kao M21a. Kvalitet rada kao i postignuti rezultati verifikovani su od strane uvaženih eksperta iz ove oblasti.

Pavlov-Kagadejev, M., Jovanovic, L., Bacanin, N., Deveci, M., Zivkovic, M., Tuba, M., Strumberger, I., Pedrycz, W.: Optimizing long-short-term memory models via metaheuristics for decomposition aided wind energy generation forecasting. Artificial Intelligence Review 57(3), 45 (2024), [94]

1.4 Sadržaj doktorske disertacije

Disertacija se sastoji iz uvoda, sedam poglavlja i zaključka.

U uvodnom delu doktorske disertacije ukratko je izložen problem koji će biti razmatran, kao i metode koje su korišćene za unapređenje rešavanja razmatranog problema. Osim navednog, u uvodnom delu prikazani su i cilj istraživanja, hipoteze, kao i metode istražanja.

Celo drugo poglavlje posvećeno je konceptu proizvodnje energije iz obnovljvih izvora. Ukratko su opisane vrste obnovljivih izvora energije, kao i njihov značaj u savremenom društvu. Poseban akcenat je na proizvodnji električne energije dobijene iz energije vetra.

S obzirom na to, da problemi koji su razmatrani u ovoj doktorskoj disertaciji spadaju u grupu optimizacionih problema, u trećem poglavlju disertacije detaljno su objašnjeni pojam i taksonomija problema optimizacije. U završnom delu ovog poglavlja ukratko su prikazane klase kompleksnosti.

Četvrto poglavlje se bavi pregledom modela mašinskog učenja, sa posebnim akcentom na modele dubokog učenja. Posebna pažnja je posvećena matematičkoj osnovi rekurentnih neuronskih mreža i podtipovima ovih mreža, koje su ispitivane za potrebe ove doktorske disertacije, kao što su: LSTM, rekurentna jedinica ograničena kapijama (eng. Gated Recurrent Unit - GRU) i bidirekcionalna LSTM (Bidirectional Long Short-Term Memory - BiLSTM). Osim navedenog, napravljen je i osvrt na jedan od najznačajnijih problema u domenu mašinskog učenja, problem optimizacije hiperparametara i mogućnostima za njegovo rešavanje.

U petom poglavlju prikazana je taksonomija optimizacionih algoritama. Poseban fokus je na kategorizaciji metaheurističkih algoritama, na metaheuristike koje nisu inspirisane prirodom i na metaheuristike insipirisane prirodom.

U šestom poglavlju, detaljnije su opisane i analizirane metaheuristike inteligencije rojeva. Uz osnovne principe po kojima ovi algoritmi funkcionišu, prikazane su

specifičnosti značajnijih predstavnika ove klase algoritama. Detaljnije su opisane optimizacija mravljim kolonijama, optimizacija rojevima čestica, optimizacija veštačkom kolonijom pčela i optimizacija svitaca.

U okviru sedmog, eksperimentalnog, poglavlja detaljno je prikazana implementacija metaheuristika i modela rekurnetnih neuronskih mreža koje su korišćene za rešavanje problema multivarijantnog predviđanja vremenskih serija proizvodnje električne energije iz energije vetra. Ovde su prikazani i skupovi podataka koji su korišćeni, i to dva skupa podataka za porizvodnju el. energije iz energije vetra.

U osmom poglavlju prikazana je statistička analiza i dati su statistički testovi koji su pokazali koji modeli su najbolji za koji tip problema. Osim toga, u ovom poglavlju prikazana je interpretacija najboljih modela, primenom SHAP analize.

U zaključku ove disertacije dati su mogući pravci daljeg razvoja i istraživanja ove oblasti.

2 Koncept proizvodnje energije iz obnovljivih izvora

Obnovljiva energija se na početku može definisati kao svaki izvor energije koji direktno, ili indirektno potiče od solarne energije. U najširem smislu, skoro sva energija koju danas koristimo, uključujući fosilna goriva, može se smatrati oblikom solarne energije. Najpoznatiji oblici energije, poput drveta, nafte, gasa i uglja, predstavljaju oblike solarne energije, koje se prirodnim procesima sakupljaju, skladište i transformišu.

Klimatske promene, koje su posledica emisije gasova sa efektom staklene bašte, naročito CO_2, postaju problem kada se prikupljena solarana energija pretvara u upotrebljive oblike energije (toplota, električna energija, goriva, hemikalije) brže nego što se formira. Za ugalj, naftu i prirodni gas, odnos vremena između formiranja i upotrebe je reda veličine milion prema jedan. Ovo znači da se za godinu dana iskoristi ono što su prirodni procesi stvarali milion godina. Jedino biomasa ima vremenski okvir koji se poklapa sa ljudskim vekom i meri se godinama, ili decenijama. Obnovljiva energija se sada može definisati kao *oblik solarne energije koji je dostupan i obnavlja se u vremenskim razmerama koje ne premašuju ljudski vek.* S obzirom na ovu definiciju obnovljive energije, postaje jasno zašto je obnovljiva energija važna opcija za ublažavanje klimatskih promena. Obnovljiva energija stvara vrlo malo (ili ni malo) emisija gasova sa efektom staklene bašte, pa se njenom upotrebom obezbeđuje održivo, dugoročno ublažavanje klimatskih promena [107].

Međunarodna agencija za obnovljivu energiju (eng. International Renewable Energy Agency - IRENA) ima zakonsku definiciju, ratifikovanu od strane 108 članova (107 država i Evropske unije) koja glasi: *obnovljiva energija obuhvata sve oblike energije proizvedene iz obnovljivih izvora na održiv način, uključujući bioenergiju, geotermalnu energiju, hidroenergiju, energiju okeana, solarnu energiju i energiju vetra* [60].

Međunarodna agencija za energetiku (eng. International Energy Agency - IEA) definiše resurse obnovljive energije *kao one koji su proizišli iz prirodnih procesa i obnavljaju se brže nego što se troše* [58].

2.1 Vrste obnovljivih izvora energije

2.1.1 Solarna Energija

Solarna tehnologija direktno koristi energiju Sunca za generisanje energije za industrijske procese, zgrade i transport, kao i za proizvodnju električne energije za opštu potrošnju [107].

Skoro svi obnovljivi izvori energije potiču isključivo od Sunca. Skoro svi obnovljivi izvori energije potiču isključivo od Sunca. Sunčevi zraci prolaze u atmosferi kroz procese apsorpcije, refleksije i transmisije na putu do Zemljine površine. Interesovanje za korišćenje solarne energije počelo je još 1970. godine, pre svega zbog tada rastućih troškova energije iz konvencionalnih izvora. Solarna radijacija je najobilniji i najtrajniji izvor energije na svetu. Količina solarne energije koju primi površina Zemlje svakog minuta, veća je od ukupne energije koju koristi celokupno stanovništvo u jednoj godini. Trenutno je solarna energija, budući da je dostupna svuda, privlačna za samostalne sisteme, posebno u ruralnim delovima zemalja u razvoju.

Precizne procene globalne solarne radijacije zahtevaju meteorološke, geografske i astronomske podatke, a posebno se mnogi modeli procene zasnivaju na lako merljivom trajanju sunčevog svetla na setu meteoroloških stanica.

Solarna energija se naziva obnovljivom i/ili održivom energijom jer će biti dostupna sve dok Sunce bude sijalo, što se procenjuje na još 4–5 milijardi godina. Energija koja potiče od Sunčeve svetlosti i stiže na površinu Zemlje, naziva se *insolacija*.

Energija vetra potiče od neujednačenog zagrevanja površine Zemlje zbog veće količine toplote na ekvatoru uz prateći transfer vode putem isparavanja i padavina. U ovom smislu, reke i brane za hidroelektričnu energiju predstavljaju skladištenje solarne energije. Treći glavni aspekt solarne energije je njeno pretvaranje u biomasu fotosintezom. Životinjski proizvodi poput kitovog ulja i biogasa iz stajskog đubriva potiču od solarne energije [108].

2.1.2 Energija Vetra

Vetroturbine pretvaraju energiju kretanja vazdušnih masa na površini Zemlje u rotiranje osovine, koja se može direktno koristiti za mehaničke energetske potrebe (npr. mlevenje ili pumpanje vode), ili konvertovati u električnu energiju pomoću

generatora. Postoje dva glavna tipa turbina, definisana na osnovu ose rotacije lopatica: sa horizontalnom osovinom (koja trenutno dominira komercijalnim tržištima) i sa vertikalnom osovinom. Energija vetra se pokazala kao najkonkurentnija tehnologija za obnovljivu električnu energiju na masovnom tržištu do sada, a njena upotreba je takođe pogodna za primenu na udaljenim mestima [107].

Vetroturbina, koja pretvara energiju vetra u električnu energiju, predstavlja jedan od najznačajnijih i najbrže razvijanih obnovljivih izvora energije širom sveta. Tehnološki razvoj, upotreba fosilnih goriva, uticaji na životnu sredinu i stalni porast konvencionalnih energetskih resursa, smanjili su troškove vetroturbina na ekonomski privlačne nivoe. Zbog toga se farme vetroturbina sve više razmatraju kao alternativni izvor energije.

Čovečanstvo je vekovima iskorišćavalo energiju vetra, kada god je bilo potrebe da se obezbedi snaga za različite zadatke. Među prvim upotrebama ove energije su prenos vode sa niže na višu nadmorsku visinu, mlevenje žitarica u mlinovima pomoću vode i sl. U poslednjih nekoliko decenija značaj energije generisane pomoću vetra, proizlazi iz njenog prijateljskog ponašanja prema životnoj sredini, koji se odnosi na zagađenje vazduha. Zbog malog zagađenja, ova energija se sve više koristi za proizvodnju električne energije, gde god je moguće, sa nadom da će zagađenje vazduha usled sagorevanja fosilnih goriva biti smanjeno. U nekim delovima SAD-a, do 20% električne energije proizvodi se energije vetra. Nakon ekonomske krize 1973. godine njegova važnost je porasla zbog ekonomskih faktora, a danas postoje vetroturbinske farme u mnogim zapadnoevropskim zemljama.

Energija vetra je pouzdana tehnologija pomoću koje se proizvodi električna energija po cenama konkurentnim sa ugljem i nuklearnom energijom [108].

2.1.3 Biomasa

Energija biomase suštinski je solarana energija koju biljke prikupe fotosintezom, a zatim je hemijski skladište. Svako gorivo koje potiče iz biomase poznato je kao biogorivo. Biomasa se može koristiti kao biogorivo direktno, na primer, kada se koristi drva kao ogrev [75].

Biomasa je termin koji obuhvata sve materijale za energiju dobijene iz bioloških izvora, uključujući otpad koji potiče od drveta, poljoprivredni otpad, otpad iz

prehrambene industrije, otpadne vode, komunalni čvrsti otpad i biljne, ili drvenaste energetske useve. Potencijalna veličina resursa biomase je prilično velika na globalnom nivou, a mogućnost korišćenja postojećih tokova otpada, koji mogu obezbediti jeftine sirovine, nudi atraktivne mogućnosti za korišćenje biomase u bliskoj budućnosti. Na duže staze, razvoj održivih, plantaža predviđenih za proizvodnju bioenergije, može dodatno proširiti resurse i pomoći u smanjenju troškova energije proizvedene iz biomase [107].

Materijal koji potiče od biljaka i životinja, uključujući njihove ostatke i otpade, je organski, ugljenični materijal koji reaguje s kiseonikom u procesima sagorevanja, pri čemu se oslobađa toplota. Takva toplota, posebno ako je na temperaturama većim od 400°C, može se koristiti za generisanje rada i električne energije. Početni materijal može se transformisati hemijskim i biološkim procesima kako bi se proizvela biogoriva, tj. biomasa prerađena u praktičniji oblik, posebno tečna goriva za transport. Primeri biogoriva uključuju metanski gas, tečni etanol, metil estre, ulja i čvrsti drveni ugalj. Pojam bioenergija se ponekad koristi kako bi obuhvatio i biomasu i biogoriva zajedno.

Drvo se može smatrati primarnim biogorivom, dok su etanol (formiran putem mikrobiološke fermentacije i destilacije useva), biogas (pretežno metan proizveden bakterijskim razgradnjom biomase) i drveni ugalj (čvrsti proizvod pirolize biomase) primeri sekundarnih biogoriva. Sva biogoriva podležu biokonverziji, što je ukupan proces u kojem se biomasa sakuplja i pretvara u gorivo, ili koristi kao gorivo [75].

Proizvodnja biomase varira s lokalnim uslovima i otprilike je dva puta veća po jedinici površine na kopnu, nego na moru.

Ugljenik u biomasi dobija se iz CO_2 u atmosferi putem fotosinteze, a ne iz fosilnih izvora. Kada se biomasa spaljuje ili razgrađuje, emitovani CO_2 se reciklira u atmosferu, ne doprinoseći povećanju koncentracije atmosferskog CO_2 tokom životnog veka rasta biomase. Energija dobijena iz biomase je iz tog razloga ugljenički neutralna. Ovo se razlikuje od korišćenja fosilnih goriva, od kojih se dodatni CO_2 emituje u atmosferu. Iz tog razloga je upotreba obnovljivih biogoriva, važan element za smanjenje emisija gasova staklene bašte. Glavne opasnosti od široke upotrebe biomase kao goriva su: seča šuma, erozija zemljišta i zamena useva hrane biljkama za gorivo. Slabo kontrolisana obrada ili sagorevanje biomase, može proizvesti neželjeno zagađenje, posebno pri relativno niskim temperaturama sagorevanja, mokrim

gorivima i nedostatku snabdevanja kiseonikom u oblastima sagorevanja. Korišćenjem održivih biogoriva umesto fosilnih goriva, smanjuje se emisija fosilnog CO_2 čime se smanjuje se uticaj na klimatske promene [121].

2.1.4 Geotermalna energija

Geotermalni energetski sistemi koriste toplotu koja potiče iz unutrašnjosti Zemlje, odnosno iz rastopljenog jezgra i raspadanja radioaktivnih materijala u kori. Potencijalna veličina ovog resursa je veoma velika. Geotermalna energija se trenutno koristi na različitim mestima širom sveta za proizvodnju električne energije po konkurentnim cenama u odnosu na konvencionalne izvore, kao i za direktno snabdevanje energijom za grejanje prostora, preradu hrane i industrijske procese, hlađenje i akvakulturu.

Tehnologije hidrotermalnih resursa, koji se sastoje od vode i/ili pare zarobljenim u pukotinama ili poroznim stenama, trenutno su jedini tip geotermalne energije koji se koristi u komercijalnom obliku. Električna energija se proizvodi putem jednog od tri glavna pravca: suva para, separisanje pare (eng. flash-steam) i binarni proces. Sistemi suve pare direktno koriste geotermalnu paru za pogon turbine-generatora, dok tehnologije separisanja pare prvo pretvaraju vruće geotermalne tečnosti u paru brzim smanjenjem pritiska. Na kraju, sistemi binarnog procesa se koriste za generisanje energije iz tečnosti niže temperature, koristeći vruće geotermalne vode za isparavanje sekundarnog radnog fluida, koji zatim pokreće jedinicu turbine-generatora.

Unutrašnje jezgro Zemlje dostiže maksimalnu temperaturu od oko 4000°C. Toplota izlazi kroz čvrstu površinu pod vodom i kopnom uglavnom putem provođenja - geotermalna toplota - a ponekad i putem aktivnih konvektivnih struja tališta magme ili zagrejane vode. Geotermalna toplota je uglavnom niskog kvaliteta i najbolje se koristi direktno za grejanje zgrada, ili procesnu toplotu na oko 50-70°C, a može se koristiti i za predzagrevanje konvencionalnih izvora visokotemperaturne energije. Takvi izvori su uspostavljeni u nekoliko delova sveta. Ponekad je geotermalna toplota dostupna na temperaturama iznad 150°C, pa se može razmatrati proizvodnja električne energije pomoću turbina.

U nekim regionima u svetu, nisko-temperaturna geotermalna energija se direktno koristi za grejanje prostora, kao što je slučaj u nekoliko gradova na Islandu gde para/topla voda nosi geotermalni fluid kroz sistem daljinskog grejanja. Još jedna

obećavajuća tehnologija je geotermalna toplotna pumpa (eng. geothermal heat pump - GTP). GTP radi kao konvencionalna toplotna pumpa (npr. frižider je toplotna pumpa jednosmerne upotrebe) i koristi temperaturni gradijent između površine Zemlje i podzemne vode, ili zemlje, nekoliko stotina stopa ispod površine za pogon pumpe. Pošto su GTP reverzibilne, mogu pružiti grejanje prostora zimi i hlađenje prostora leti, kao i dodatnu podršku potrebi za toplom vodom tokom cele godine [107, 121].

2.1.5 Hidroenergija

Postrojenja za hidroenergiju iskorišćavaju kinetičku energiju tekuće, ili padajuće vode kako bi generisala električnu energiju. Konvencionalna hidroenergetska postrojenja koriste vodu iz reka, potoka, kanala, ili rezervoara kako bi kontinualno proizvodila električnu energiju [121].

Hidroenergija je već uspostavljen tehnološki način proizvodnje obnovljive energije. U zemljama sa industrijskim razvojem i bogatim površinskim vodama, potpuni razvoj proizvodnje hidroelektrične energije putem turbina na velikim branama, već je iskorišćen do krajnjih granica, te su stoga manji hidro-sistemi interesantni kako bi se pristupilo marginalnim resursima [108].

Kao rezultat klimatskih promena do 2070-ih godina, očekuje se da će potencijal hidroenergije za celu Evropu opasti za 6%, pru čemu se očekuje da se smanji za 20–50% oko Sredozemlja, poveća za oko 15–30% na severu i istoku Evrope, dok se u Zapadnoj i Centralnoj Evropi ne očekuju veće promene proizvodnje hidroenergije.

Mini-hidro postrojenja (veličine 30 MW ili manje) nude prilike za distribuiranu ili udaljenu proizvodnju el. energije čiji je uticaj na životnu sredinu minimalan, operativni troškovi niski, a pouzdanost visoka. Instalacija ovih sistema obično je veoma brza i može koristiti lokalnu radnu snagu. Hidroenergetska tehnologija je trenutno široko dostupna. Samo deo dostupnih resursa je iskorišćen do sada, uglavnom zbog ograničenja lokacije, pritisaka zaštite životne sredine protiv sistema velikog obima i konkurencije sa drugim interesima za upotrebu vodnih resursa [121].

Hidroelektrane i postrojenja mogu biti dugovečni uz rutinsko održavanje, npr. turbine traju oko pedeset godina i duže uz manje obnove, a brane i do sto godina. Dug vek trajanja turbine potiče od neprekidnog, stabilnog rada bez visoke temperature,

ili drugih stresova. Iz tog razloga ova postrojenja često proizvode električnu energiju po niskoj ceni.

Glavni nedostaci hidroenergije uključuju moguće nepovoljne ekološke uticaje, efekat na ribe, zasipanje brana, koroziju turbina u određenim vodenim uslovima, društvene uticaje raseljavanja ljudi sa mesta rezervoara, gubitak potencijalno produktivnog zemljišta i relativno visoke kapitalne troškove u poređenju s troškovima postrojenja na fosilna goriva [107].

2.1.6 Energija okeana

Okean je najveći solarni kolektor na svetu. U tropskim morima, mogu se javiti temperaturne razlike od oko 20-25°C između tople, površinske vode koja apsorbuje sunčevu energiju i hladnije vode na dubini od 500–1000 m [121]. Da bi se iskoristila energija talasa, plime i oseke okeana, osmišljeni su mnogi sistemi.

Najveću pažnju su do sada privukli OTEC (eng. Ocean Thermal Energy Conversion) i sistemi za plimu i oseku. OTEC sistemi koriste temperaturnu razliku između površine okeana i dubina do 1000 metara kako bi generisali električnu energiju: u zatvorenom ciklusu (koristeći sekundarni radni fluid), ili u otvorenom ciklusu (koristeći morsku vodu kao radni fluid).

Da bi se dobila energija iz plime i oseke, koristi se isti princip koji se koristi u generisanju hidroelektrične energije. Energija se izdvaja iz razlike u hidrostatičkom pritisku, koji nastaje usled plime i oseke. Kao potrebna granica, da bi se efikasno proizvela električna energija, često se navodi minimalna razlika od 5 metara između plime i oseke, iako razvoji u mikro-hidro tehnologiji mogu omogućiti pristup dodatnim resursima sa nižim opsegom. OTEC i energija plime i oseke su i dalje uglavnom u fazi razvoja, pri čemu tehnička, ekonomska i lokacijska ograničenja ograničavaju napredak ovih sistema [107].

2.2 Proizvodnja električne energije iz energije vetra

Vetrogenerator, ili vetroturbina, je uređaj koji pretvara snagu vetra (kinetičku energiju vetra) u električnu energiju. Kao generatori električne energije, vetrogeneratori su povezane sa električnom mrežom. Najčešće se sreću vetrogeneratori malih snaga - reda veličine do 10 kW, ili manje. Što se tiče ukupnog kapaciteta proizvodnje,

većinu čine vetrogeneratori u rasponu od 1.5 do 5 MW. Veće turbine se uglavnom koriste u velikim javnim mrežama.

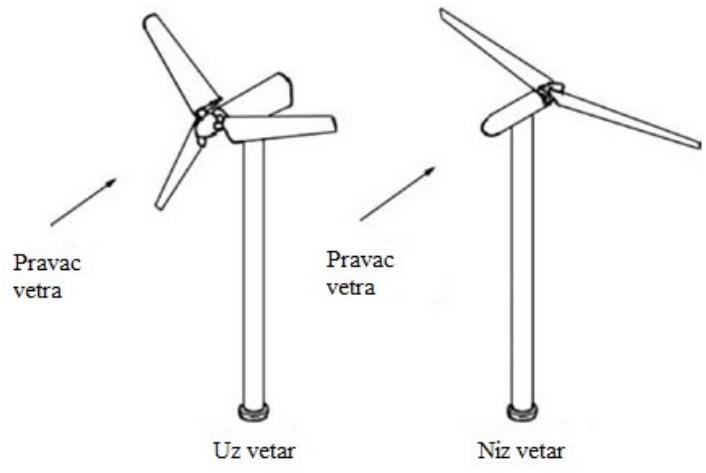
Vetrogeneratori, za razliku od većine drugih generatora, mogu proizvoditi energiju samo kao odgovor na trenutno dostupan izvor energije. Nemoguće je skladištiti vetar i koristiti ga kasnije. Iz tog razloga izlaz iz vetrogeneratora je promenljiv i nepredvidiv, pa svaki sistem sa kojim je vetrogenerator povezan mora, na neki način, uzeti u obzir ovu varijabilnost. U većim mrežama, vetrogenerator služi da smanji ukupno električno opterećenje i na taj način se smanjuje broj konvencionalnih generatora koji se koriste ili se smanje potrošnja goriva kod onih koji su u pogonu. U manjim mrežama, mogu postojati skladišta energije, rezervni generatori i neki specijalizovani kontrolni sistemi. Još jedna činjenica je da vetar nije prenosiv: može se pretvoriti samo tamo gde duva. Istoriski gledano, proizvod poput samlevenog žita je pravljen na mestu na kome se nalazila vetrenjača, a zatim prevožen na mesto upotrebe. Danas, mogućnost prenosa električne energije putem dalekovoda donekle kompenzuje nemogućnost da se vetar prenese na drugu lokaciju [81].

2.2.1 Dizajn modernih vetrogeneratora

Najčešći dizajn vetrogeneratora je vetrogenerator sa horizontalnom osom (eng. Horizontal Axis Wind Turbine - HAWT), kod koga je osa rotacije paralelna sa zemljom. Rotori HAWT-a se obično klasifikuju prema orijentaciji, dizajnu glavčine, kontroli rotora, broju lopatica i načinu na koji su usklađeni sa vetrom. Slika 1 prikazuje dve konfiguracije vetrogeneratora zavisno od smera vetra.

Glavni podsistemi tipičnog vetrogeneratora sa horizontalnom osom koji su smešteni na kopnu su:

- Rotor, koji se sastoji od lopatica i potpomažuće glavčine.
- Sistem za prenos snage, koji obuhvata rotirajuće delove vetroturbine (isključujući rotor); obično se sastoji od osovina, menjača, spojnica, mehaničke kočnice i generatora.
- Kućišta i glavnog okvira, uključujući kućište vetroturbine, nosač i sistem okretanja.
- Tornja i temelja.
- Kontrole mašine.



Slika 1: *Konfiguracija HAWT rotora [81]*

- Ostatak električnog sistema, uključujući kablove, prekidače, transformatore i eventualno elektronske pretvarače snage.

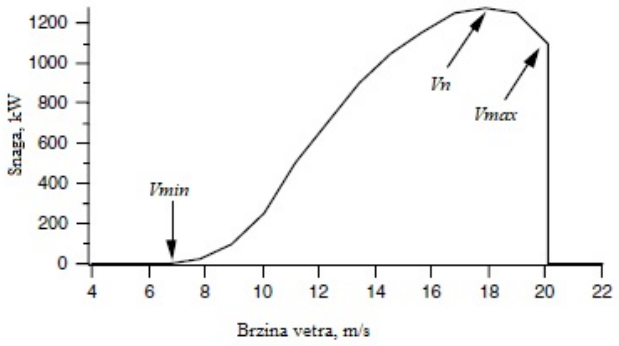
Pored HAWT turbina, predložen je i veliki broj drugih topologija, međutim ni jedna od njih nije postigla isti stepen uspeha, kao HAWT turbina. Najkonkurentnija je bila Darrieusova turbina sa vertikalnom osom (eng. Vertical Axis Wind Turbine - VAWT). Ovi vetrogeneratori su proučavani u SAD-u i Kanadi 70-tih i 80-tih godina, međutim imali su ozbiljnih problema kada je u pitanju pouzdanost.

2.2.2 Predviđanje izlazne snage

Izlazna snaga vetrogeneratora varira sa brzinom vetra, a svaka vetroturbina ima karakterističnu krivu snage. Pomoću ove krive moguće je predvideti proizvodnju energije vetroturbine, ne uzimajući u obzir tehničke detalje njenih različitih komponenti. Kriva snage daje električni izlaz snage kao funkciju brzine vetra na visini glavčine. Na Slici 2 je prikazana tipična kriva snage za vetroturbinu.

Performanse vetrogeneratora zavise od tri ključne vrednosti koje se odnose na brzinu:

- V_{min} je brzina početka rada. To je minimalna brzina vetra pri kojoj može da se proizvodi korisna snaga.
- V_n je nominalna brzina vetra. Predstavlja brzinu vetra pri kojoj se dostiže nominalna snaga. Obično je to maksimalna izlazna snaga električnog generatora.
- V_{max} je brzina prekida. To je maksimalna brzina vetra pri kojoj je dozvoljeno da



Slika 2: *Tipična kriva snage vetrogeneratora*

turbina proizvodi snagu, što je obično ograničeno inženjerskim dizajnom i sigurnosnim ograničenjima.

Krive snage se obično mogu dobiti od proizvođača. One se formiraju na osnovu terenskih ispitivanja, korišćenjem standardizovanih metoda ispitivanja.

2.2.3 Karakteristike i resursi vetra

Poznavanje karakteristika vetra na određenoj lokaciji ima veliki značaj prilikom:

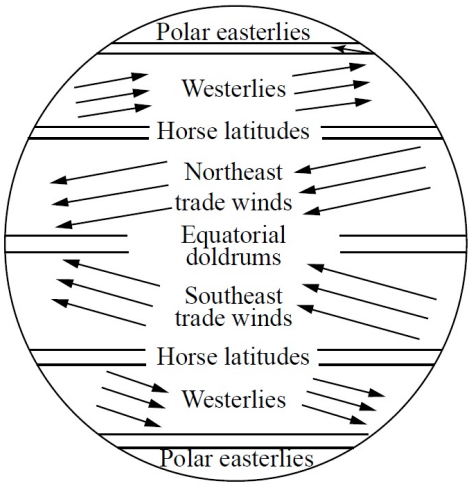
- Dizajna sistema - pri dizajnu sistema neophodno je poznavati reprezentativne prosečne uslove vetra, imati informacije o turbulentnoj prirodi vetra, kao i o ekstremnim vetrovima.
- Evaulacije performansi - podrazumeva određivanje očekivane energetske produktivnosti i ekonomske efikasnosti određenog sistema za energiju vetra, na osnovu resursa vetra.
- Odabira lokacije.

2.2.4 Globalno poreklo vetra

Globalni vetrovi nastaju zbog razlika u pritisku na površini Zemlje usled neujednačenog zagrevanja Zemlje Sunčevim zračenjem. Na primer, količina Sunčevog zračenja apsorbovana na površini Zemlje je veća na ekvatoru nego na polovima. U nižim slojevima atmosfere (troposferi), varijacije dolazne energije stvaraju konvektivne ćelije. To su oblasti unutar atmosfere gde je proces vertikalnog kretanja vazduha intenzivniji. U osnovnom konceptu atmosferske cirkulacije, vazduh se podiže na ekvatoru i spušta na polovima. Cirkulacija atmosfere koja proizlazi iz neujednačenog

zagrevanja pod snažnim je uticajem efekata koji su posledica rotacije Zemlje (brzina od oko 1670 km/h na ekvatoru, opada do nule na polovima). Pored toga, sezonske varijacije u distribuciji solarnog zračenja dovode do varijacija u cirkulaciji.

Prostorne varijacije u prenosu toplote u atmosferu Zemlje stvaraju varijacije atmosferskog pritiska koje uzrokuje kretanje vazduha od područja visokog ka područjima niskog pritiska. U vertikalnom pravcu postoji sila gradijenta pritiska, ali je ova sila obično neutralisana gravitacionom silom. Vetrovi uglavnom duvaju u horizontalnoj ravni, reagujući na horizontalne gradijente pritiska. Istovremeno, postoje sile koje teže mešanju vazdušnih masa, različitih temperatura i pritiska raspoređenih po površini Zemlje. Osim gradijenta pritiska i gravitacione sile, inercija vazduha, rotacija Zemlje i trenje o površinu Zemlje (koje rezultira turbulencijom) utiču na atmosferske vetrove [81].



Slika 3: *Cirkulacija površinskih vetrova [81]*

Na Slici 3 prikazana je cirkulacija površinskih vetrova širom planete. Ovaj model je pojednostavljen, jer ne uzima u obzir efekte koji imaju kopnene mase na raspodelu vetra.

2.2.5 Mehanika kretanja vetra

U jednom od najjednostavnijih modela za mehaniku kretanja vetra u atmosferi, mogu se razmatrati četiri atmosferske sile. To uključuje sile pritiska, Koriolisovu silu (eng. Coriolis force) izazvanu rotacijom Zemlje, inercijalne sile usled velikih kružnih pokreta i sile trenja na površini Zemlje.

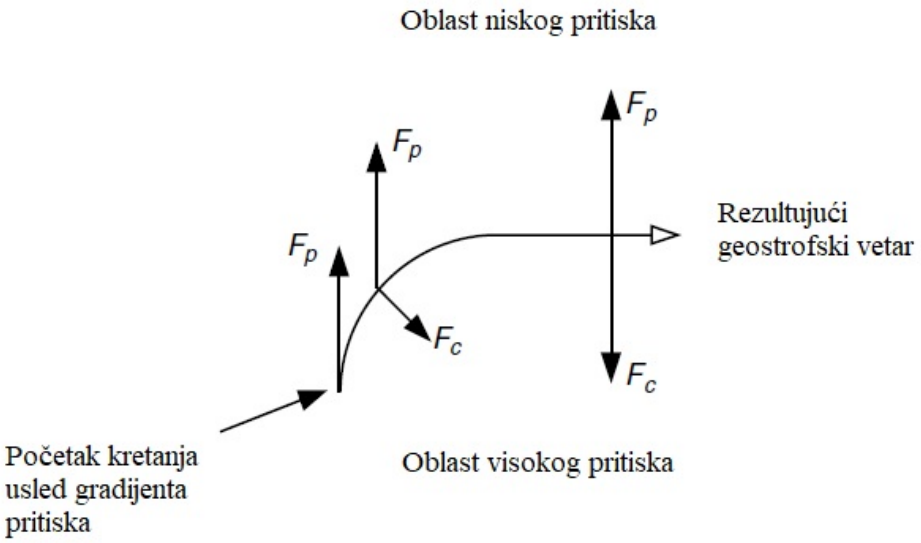
Sila pritiska na vazduh (po jedinici mase) F_p, data je sledećom formulom:

$$F_p = \frac{-1}{\rho}\frac{\partial p}{\partial n} \tag{1}$$

gde je r gustina vazduha, a n pravac normalan na linije konstantnog pritiska. Odnos $\partial p/\partial n$ je definisan kao gradijent pritiska normalan na linije konstantnog pritiska, ili izobare. Koriolisova sila (po jedinici mase) F_c, je fiktivna sila koja se izražava se kao:

$$F_c = fU \tag{2}$$

gde je U brzina vetra, a f je Koriolisov parametar $[f = 2cos\omega(\varphi)]$. φ predstavlja geografsku širinu, a ω je ugaona rotacija Zemlje. Dakle, veličina Koriolisove sile zavisi od brzine vetra i geografske širine. Smer Koriolisove sile je normalan na smer kretanja vazduha. Rezultujuća sila ove dve sile, naziva se *geostrofski vetar* i ima tendenciju da bude paralelna izobarama (Slika 4).



Slika 4: *Ilustracija geostrofskog vetra*

Veličina (magnituda) geostrofskog vetra U_g, je funkcija ravnoteže sila i data je sledećom formulom:

$$U_g = \frac{-1}{f\rho}\frac{\partial p}{\partial n} \tag{3}$$

Ovo je idealizovan slučaj budući da prisustvo područja visokog i niskog pritiska uzrokuje zakrivljenost izobara. To nameće dodatnu, centrifugalnu silu. Rezultirajući vetar nazvan je *gradijentni vetar* U_{gr}. Gradijentni vetar je takođe paralelan izobarama

i rezultat je ravnoteže sila:

$$\frac{U_{gr}^2}{R} = -fU_{gr} - \frac{1}{\rho}\frac{\partial p}{\partial n} \tag{4}$$

gde je R poluprečnik krivine puta vazdušnih čestica, a zamenom vrednosti U_g, dobijamo

$$U_g = U_{gr} + \frac{U_{gr}^2}{fR} \tag{5}$$

Poslednja sila koja deluje na vetar je posledica trenja sa površinom Zemlje. Površina Zemlje deluje horizontalnom silom na vazduh koji se kreće, čiji je efekat usporavanje protoka. Ova sila se smanjuje sa povećanjem visine iznad Zemlje i postaje zanemarljiva iznad granice sloja trenja (definisanog kao bliska regija atmosfere gde su viskozne sile značajne). Iznad graničnog sloja, uspostavlja se balans vetra bez trenja i vetar se kreće brzinom gradijentnog vetra duž izobara. Trenje na površini uzrokuje skretanje vetra više prema regiji niskog pritiska.

2.2.6 Ostali oblici atmosferske cirkulacije

Površina Zemlje nije glatka, već je raznolika, sa velikim okeanima i kopnenim površinama. Okeani deluju kao veliki rezervoar energije, pa je kretanje vazduha često pod uticajem njihove cirkulacije. Svi ovi efekti dovode do diferencijalnih pritisaka koji utiču na globalne vetrove i mnoge regionalne vetrove koji se javljaju tokom monsuna. Osim toga, lokalno zagrevanje, ili hlađenje, može izazvati pojavu postojanih lokalnih vetrova na sezonskoj ili dnevnoj osnovi. To uključuje morske povetarce i planinske vetrove. Cirkulacija atmosfere manjih razmera može se podeliti na sekundarnu i tercijarnu cirkulaciju.

Sekundarna cirkulacija se javlja ako su centri visokog ili niskog pritiska uzrokovani zagrevanjem, ili hlađenjem niže atmosfere. Sekundarne cirkulacije uključuju: uragane, monsunsku cirkulaciju i ekstratropske ciklone.

Tercijarne cirkulacije su lokalne cirkulacije malih razmera koje karakterišu lokalni vetrovi, kao što su: vetrovi sa kopna i mora, dolinski i planinski vetrovi, struja slična monsunima, fen vetrovi, oluje s grmljavinom i tornada.

2.3 Prostorne i vremenske karakteristike vetra

Atmosferska kretanja variraju kako u vremenu tako i u prostoru. Prostorne varijacije generalno zavise od visine iznad Zemlje na kojoj se odvija atmosferska cirkulacija i od globalnih i lokalnih geografskih uslova.

2.3.1 Varijacija u vremenu

Varijacije brzine vetra u vremenu mogu se podeliti u sledeće kategorije:

- međugodišnje,
- godišnje,
- dnevne i
- kratkoročne (udari i turbulencija).

Međugodišnje varijacije brzine vetra

Međugodišnje varijacije u brzini vetra se javljaju na vremenskim skalama dužim od jedne godine. One mogu imati veliki uticaj na dugoročnu proizvodnju vetrotgeneratora. Mogućnost procene međugodišnje varijabilnosti na određenoj lokaciji je važna kao i procena dugoročne prosečne brzine vetra na toj lokaciji. Meteorolozi generalno zaključuju da je potrebno 30 godina podataka, da bi se formirale dugoročne prognoze vremena ili klime, dok je minimum 5 godina neophodno kako bi se dobila pouzdana prosečna godišnja brzina vetra na određenoj lokaciji. Ipak, kraći vremenski zapisi mogu biti korisni. Jedno statistički razvijeno pravilo je: jedna godina podataka obično je dovoljna da se predvide dugoročne sezonske prosečne brzine vetra sa tačnošću od 10%, uz nivo poverenja od 90%. Istraživači još uvek traže pouzdane modele za dugoročno predviđanje prosečne brzine vetra. Kompleksnost interakcija meteoroloških i topografskih faktora koji uzrokuju njenu varijaciju otežava ovaj zadatak.

Godišnje varijacije brzine vetra

Značajne varijacije u sezonskim, ili mesečnim prosečnim brzinama vetra su česte širom sveta. Na primer, za istočnu trećinu Sjedinjenih Američkih Država, maksimalne brzine vetra se javljaju tokom zime i ranog proleća. Maksimumi u proleće se javljaju na velikim ravnicama, u severnim centralnim državama, na obali Teksasa, u basenima i dolinama Zapada, kao i na obalnim područjima Srednje i Južne Kalifornije. Zimski maksimumi se javljaju u svim planinskim regionima SAD, osim nekih područja u jugozapadnom delu gde se javljaju maksimumi u proleće. Maksimumi u proleće i leto

se javljaju u koridorima vetra u Oregonu, Vašingtonu i Kaliforniji.

Dnevne varijacije brzine vetra

Na tropskim i umerenim geografskim širinama, mogu se javiti velike varijacije vetra na dijurnoj ili dnevnoj skali. Ovaj tip varijacije brzine vetra je rezultat diferencijalnog zagrevanja površine Zemlje tokom dnevnog ciklusa zračenja. Tipična dnevna varijacija je povećanje brzine vetra tokom dana, pri čemu su najniže brzine vetra u periodu od ponoći do izlaska sunca. Dnevne varijacije u solarnom zračenju su odgovorne za dnevne varijacije vetra na umerenim geografskim širinama nad relativno ravnom površinom zemlje. Najveće dnevne promene se obično javljaju u proleće i leto, a najmanje zimi. Osim toga, dnevna varijacija brzine vetra može se razlikovati u zavisnosti od lokacije i nadmorske visine. Na primer, na većim nadmorskim visinama u odnosu na okolni teren, kao što su planine ili grebeni, dnevni obrazac može biti veoma različit. Iako se osnovne karakteristike dnevnog ciklusa mogu utvrditi na osnovu jedne godine podataka, detaljnije karakteristike kao što su amplituda dnevne oscilacije i vreme dana kada se javljaju maksimalni vetrovi, ne mogu se odrediti precizno.

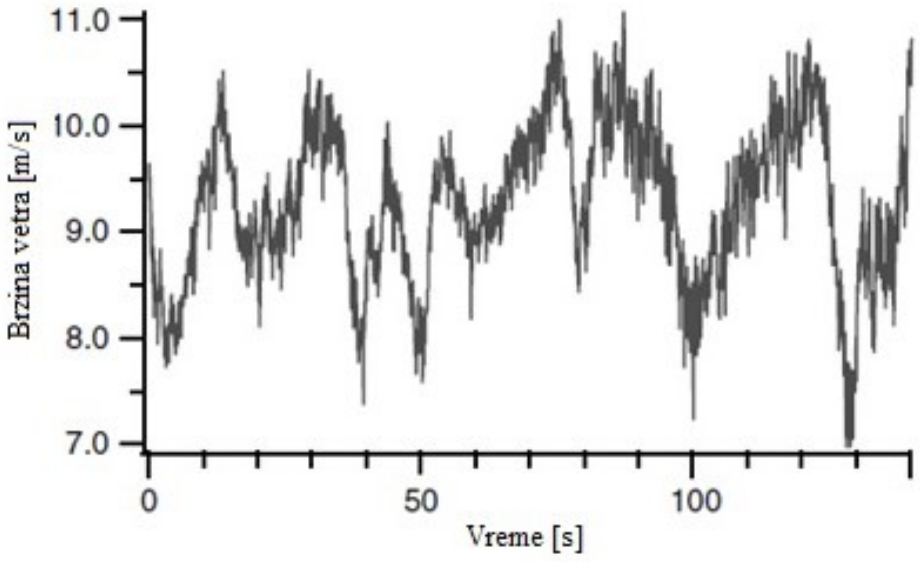
Kratkoročne varijacije brzine vetra

Kratkoročne varijacije brzine vetra koje su od interesa uključuju turbulenciju i udare vetra. Slika 5 , koja je rezultat merenja anemometrom, prikazuje tipične kratkoročne varijacije brzine vetra. Kratkoročne varijacije obično podrazumevaju varijacije tokom vremenskih intervala od deset minuta ili manje. Prosečne vrednosti u intervalima od deset minuta obično se određuju uzorkovanjem brzine vetra sa frekvencijom uzorkovanja od oko 1 sekunde. Opšte je prihvaćeno da varijacije brzine vetra sa periodima manjim od jedne sekunde do deset minuta, koje imaju stohastički karakter, predstavljaju turbulenciju.

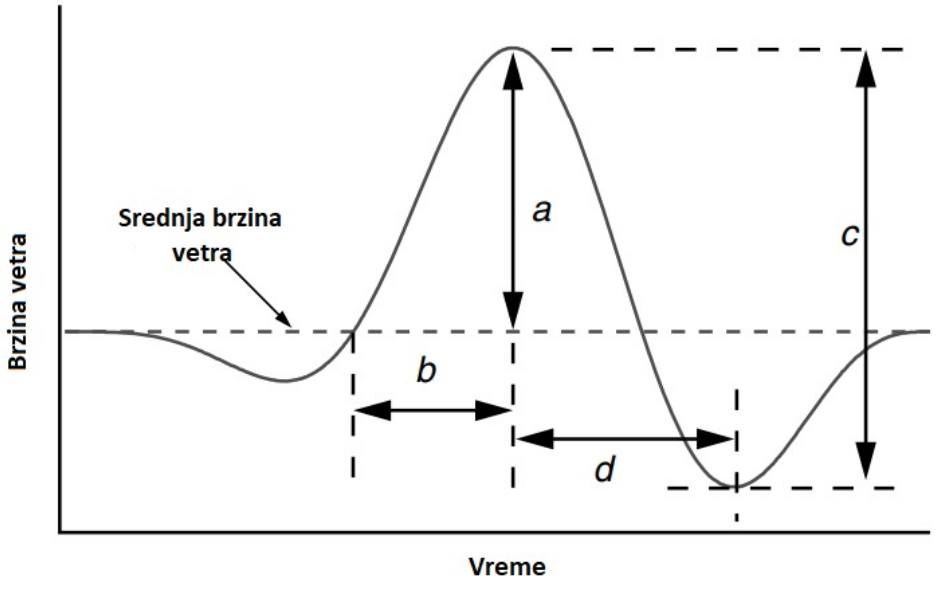
Da bi se primenile za proizvodnju energije, potrebno je kvantifikovati turbulenciju u strujanju vetra. Turbulencija se može shvatiti kao nasumične fluktuacije brzine vetra nametnute srednjoj brzini vetra. Ove fluktuacije se javljaju u sva tri pravca: uzdužno (u smeru vetra), bočno (normalno na prosečni vetar) i vertikalno.

Udar vetra je diskretni događaj unutar turbulencije vetra. Jedan od načina za karakterizaciju udara je određivanje: (a) amplitude, (b) vremena porasta, (c) maksimalne varijacije udara i (d) vremena pada (Slika 6). Opterećenja na vetro-

generatorima, koja su izazvana udarima vetra su pod uticajem ova četiri faktora.



Slika 5: *Tipičan dijagram promene brzine vetra sa vremenom u kratkom periodu*



Slika 6: *Diskretni udari vetra: (a) amplituda; (b) vreme uzdizanja; (c) maksimalna varijacija naleta vetra; (d) vreme opadanja.*

2.3.2 Varijacije u zavisnosti od lokacije i smera vetra

Brzina vetra takođe zavisi od lokalnih topografskih i varijacija na Zemljinom omotaču. Razlike u brzini vetra između dva mesta koja međusobno nisu mnogo udaljena, mogu biti značajne.

Smer vetra varira u istim vremenskim razmacima tokom kojih se menja brzina vetra. Sezonske varijacije mogu biti male, oko 30 stepeni, ili prosečni mesečni vetrovi mogu menjati smer za 180 stepeni tokom godine. Kratkoročne promene smera su rezultat turbulentne prirode vetra. Ove kratkotrajne varijacije u smeru vetra moraju se uzeti u obzir pri dizajniranju i postavljanju vetrogeneratora. Vetrogeneratori sa horizontalnom osom moraju se rotirati sa promenama u smeru vetra. Rotacija izaziva žirospopska opterećenja kroz celokupnu strukturu turbine i aktivira sve mehanizme uključene u rotaciono kretanje. Poprečni vetrovi usled promena u smeru vetra utiču na opterećenje lopatica. Kratkotrajne varijacije u smeru vetra i kretanje usled njih utiču na zamor materijala komponenti kao što su lopatice i mehanizmi za rotaciju.

3 Optimizacioni problemi

Optimizacija je grana primenjene matematike i matematički gledano, optimizacija se odnosi na minimizaciju (ili maksimizaciju) date funkcije cilja više promenljivih za donošenje odluka koje zadovoljavaju ograničenja funkcije.

Promenljive odlučivanja, funkcija cilja i ograničenja su tri osnovna elementa svakog problema optimizacije. Problemi koji nemaju ograničenja nazivaju se *optimizacioni problemi bez ograničenja* (eng. unconstrained optimization), dok se problem sa ograničenjima često nazivaju *optimizacioni problem sa ograničenjima* (eng. constrained optimization). Problemi bez funkcija cilja nazivaju se *problemima izvodljivosti* (eng. feasibility problems). Neki problemi mogu imati više funkcija cilja i oni se često rešavaju svođenjem na probleme sa jednom funkcijom cilja, ili niz takvih problema.

Ako su promenljive odlučivanja u problemu optimizacije ograničene celim brojevima ili diskretnim skupom mogućnosti, imamo problem *celobrojne* (eng. integer) ili *diskretne optimizacije* (eng. discrete optimization problem). Ako ne postoje takva ograničenja za promenljive, radi se o *kontinualnom problemu optimizacije* (eng. continuous optimization problem). Neki optimizacioni problemi mogu sadržati obe vrste promenljivih, diskretne i kontinualne [30].

Optimizacija je važan proces koji se koristi u naukama o odlučivanju i analizi fizičkih sistema. Da bi se optimizacija iskoristili na pravi način, prvo je potrebno identifikovati neki cilj, kvantitativnu meru performansi sistema koji se proučava. Taj cilj može biti profit, vreme, potencijalna energija, ili bilo koja veličina koja se može predstaviti jednim brojem. Cilj zavisi od određenih karakteristika sistema, nazvanih promenljive ili nepoznate. Iz tog razloga je potrebno pronaći vrednosti promenljivih koje optimizuju cilj. Često su promenljive ograničene na neki način, npr. neke veličine ne mogu biti negativne.

Proces identifikacije cilja, promenljivih i ograničenja za dati problem, poznat je kao *modeliranje*. Prvi korak u procesu optimizacije je izrada odgovarajućeg modela. Ako je model previše pojednostavljen, neće pružiti koristan uvid u praktični problem, a ako je previše kompleksan, može biti previše težak za rešavanje.

Nakon formulisanja modela, algoritam za optimizaciju se može koristiti za pronalaženje rešenja. Za optimizaciju se ne koristi jedan algoritam, već skup algoritama, od kojih svaki odgovara određenoj vrsti problema optimizacije. Odgovornost za izbor

odgovarajućeg algoritma za određenu primenu često leži na korisniku. Taj izbor je važan, jer može odrediti brzinu rešavanja problema, pa čak da li će rešenje uopšte biti pronađeno.

Nakon što je algoritam za optimizaciju primenjen na model, potrebno je prepoznati da li je pronašao odgovarajuće rešenje. *Uslovi optimalnosti* (eng. optimality conditions) predstavljaju matematičke izraze i koriste se kako bi se izvršila provera da li aktuelni skup promenljivih predstavlja rešenje problema. U slučaju da uslovi optimalnosti nisu zadovoljeni, oni ipak pružaju informacije kako trenutna procena rešenja može postati bolja. Za unapređenje modela može se primeniti tehnika poznata kao *analiza osetljivosti* (eng. sensitivity analysis). Ova tehnika pokazuje koliko promene u modelu i podacima utiču na rešenja. Tumačenje rešenja u smislu primene, takođe može sugerisati načine na koje se model može unaprediti ili poboljšati. Ako se bilo kakve promene naprave na modelu, problem optimizacije se rešava iznova, i proces se ponavlja [93].

U optimizacionom problemu P, imamo skup X koji predstavlja prostor rešenja i funkciju cilja $f : X \to R$. Cilj je pronaći optimalno rešenje u skupu X u skladu sa funkcijom cilja f.

Za probleme minimizacije, treba pronaći rešenje $x^* \in X$:

$$f(x^*) \leq f(x), \quad \forall x \in X \tag{6}$$

Za probleme maksimizacije, treba pronaći rešenje $x^* \in X$:

$$f(x^*) \geq f(x), \quad \forall x \tag{7}$$

Svako rešenje x u P je određeno skupom vrednosti koje su dodeljene promenljivim odlučivanja. Moguće vrednosti su ograničene skupom ograničenja, često izraženih u obliku jednakosti, nejednakosti ili logičkih izraza. Kako bismo pronašli rešenje x sa najboljom mogućom vrednošću $f(x)$, moraju se razviti tačni ili heuristički postupci optimizacije koji se koriste za pronalaženje optimalnih, ili rešenja koja su bliska optimalnim rešenjima [85].

3.1 Podela optimizacionih problema

Problemi optimizacije se mogu klasifikovati prema prirodi funkcije cilja i ograničenjima (linearnim, nelinearnim, konveksnim), broju promenljivih (velikim ili malim), glatkosti funkcija (diferencijabilnim ili nediferencijabilnim) i tako dalje [93]. Optimizacija se generalno može kategorisati kao diskretna ili kontinualna, zavisno od tipa promenljivih. Mogu postojati ograničenja ili uslovi na promenljivama. Optimizacija može biti statički ili dinamički problem, u zavisnosti od toga da li je izlaz funkcija vremena [38].

3.1.1 Podela optimizacionih problema prema tipu varijabli

Kod nekih optimizacionih problema promenljive imaju smisla samo ako uzimaju celobrojne vrednosti. Matematička formulacija takvih problema uključuje celobrojna ograničenja, koja imaju oblik $x_i \in Z$, gde je Z skup celih brojeva, ili binarna ograničenja, koja imaju oblik $x_i \in \{0, 1\}$. Problemi ove vrste nazivaju se *problemima celobrojnog programiranja* (engl. Integer Programming Problems). Ako neke promenljive u problemu nemaju ograničenje da moraju biti celobrojne ili binarne promenljive, ponekad se nazivaju *problemima mešaovitog celobrojnog programiranja* (eng. Mixed Integer Programming Problems - MIP). U zavisnosti od tipa varijabli, optimizacione probleme možemo da svrstamo u dve kategorije : diskretni optimizacioni problemi i kontinualni optimizacioni problemi.

Problemi celobrojnog programiranja su vrsta *problema diskretne optimizacije*. Uopšteno, problemi diskretne optimizacije mogu sadržati ne samo cele brojeve i binarne promenljive, već i apstraktne promenljive objekte poput permutacija uređenog skupa. Ključna karakteristika problema diskretne optimizacije je da je nepoznata x izvučena iz konačnog (ali često vrlo velikog) skupa. Nasuprot tome, skup dopuštenih vrednosti za *probleme kontinualne optimizacije*, obično je nebrojivo beskonačan, kao u slučaju kada se komponentama x dopušta da budu realni brojevi. Problemi kontinualne optimizacije obično su lakši za rešavanje [93].

Diskretni optimizacioni problemi

Diskretni optimizacioni problemi poznati su i kao *problemi kombinatorne optimizacije* (eng. Combinatorial Optimization Problem - COP). Svaki problem koji ima

veliki skup diskretnih rešenja i funkciju troška za ocenu tih rešenja spada u COP. Kombinatorni optimizacioni problemi spadaju u NP-teške probleme. Cilj diskretne optimizacije je pronaći optimalno rešenje, ili rešenje koje je približno optimalnom. Kod ove vrste optimizacionih problema, broj rešenja eksponencijalno raste sa veličinom problema n u $O(n!)$, ili $O(e^n)$, tako da ni jedan algoritam ne može pronaći globalno minimalno rešenje u polinomskom vremenu računanja.

Definicija 1.1 Problem diskretne optimizacije označava se kao (X, f, Ω), ili kao minimizacija funkcije cilja

$$minf(x), \quad x \in X, \quad uz \quad ograničenja \quad \Omega \tag{8}$$

gde je $X \subset R^N$ prostor pretrage definisan nad konačnim skupom N diskretnih promenljivih odlučivanja $x = (x_1, x_2, ..., x_N)^T$, $f : X \to R$, dok Ω predstavlja skup ograničenja na x. Prostor X formira se u skladu sa svim ograničenjima koja su zadata za dati optimizacioni problem.

Vektor x koji zdovoljava skup ograničenja za optimizacioni problem, naziva se prihvatljivo rešenje (eng. feasible solution).

Problem trgovačkog putnika (eng. Traveling Salesman Problem - TSP) predstavlja jedan od najpoznatijih problema kombinatorne optimizacije [38].

Kontinualni optimizacioni problemi

U kontinualnoj optimizaciji, promenljive u modelu nominalno mogu uzeti kontinualni niz vrednosti, obično realnih brojeva. Ova karakteristika razlikuje kontinualnu optimizaciju od diskretne ili kombinatorne optimizacije, gde promenljive mogu biti binarne (ograničene na vrednosti 0 i 1), celobrojne (gde su dozvoljene samo celobrojne vrednosti), ili apstraktni objekti iz skupova sa konačnim brojem elemenata.

Problemi kontinualne optimizacije obično se rešavaju pomoću algoritama koji generišu niz vrednosti promenljivih, poznatih kao iteracije, koje konvergiraju ka rešenju problema. Pri odlučivanju kako preći sa jedne iteracije na sledeću, algoritam koristi saznanja stečena na prethodnim iteracijama i informacije o modelu u trenutnoj iteraciji, uključujući informacije o njegovoj osetljivosti na promene promenljivih [128].

3.1.2 Podela optimizacionih problema u zavisnosti od ograničenja

U praksi, optimizacioni problemi su veoma često ograničeni, dok je neograničenih optimizacionih problema malo. Jedan primer neograničenog optimizacionog problema je prilagođavanje podataka, gde se prilagođava kriva na izmerenim podacima. Ovo se postiže odgovarajućim modifikacijama funkcije cilja, uključujući kaznene izraze, u slučaju kršenja ograničenja.

Metode rešenja za neograničene optimizacione probleme mogu se uopšteno klasifikovati na metode zasnovane na gradijentu i metode pretrage koje ne koriste gradijent. Kao što naziv sugeriše, metode zasnovane na gradijentu zahtevaju informacije o gradijentu pri određivanju smera pretrage. U metode zasnovane na gradijentu spadaju: metod najstrmijeg spusta (eng. Steepest Descent Method), U metode zasnovane na gradijentu spadaju: metod najstrmijeg spusta (eng. Steepest Descent Method), DFP (Davidon–Fletcher–Powell), BFGS (Broyden–Fletcher–Goldfarb–Shanno), Njutnov i Levenberg–Marquardt metod. U metode pretrage bez gradijenta spadaju Powell-ova metoda konjugovanih pravaca i Simplex metoda (Nelder–Mead Algoritam) [9].

Problemi sa ograničenjima potiču iz modela u kojima ograničenja igraju ključnu ulogu. Ova ograničenja mogu biti jednostavna poput $0 \leq x_1 \leq 100$, opširnija linearna ograničenja poput $\sum_i x_i \leq 1$, ili nelinearne nejednakosti koje predstavljaju kompleksne odnose među promenljivima. Kada su funkcija cilja i sva ograničenja linearne funkcije, reč je o *problemima linearnog programiranja*. Problemi ovog tipa verovatno su najšire postavljani i rešavani od svih problema optimizacije, posebno u aplikacijama upravljanja, finansija i ekonomije. *Problemi nelinearnog programiranja*, u kojima su bar neka od ograničenja ili funkcija cilja nelinearni, javljaju se u prirodnim i tehničkim naukama, i postaju sve više korišćeni i u naukama o upravljanju i ekonomiji [93].

Optimizacioni problemi sa ograničenjima

Nelinearno programiranje (ili optimizacija sa ograničenjima) bavi se problemom optimizacije funkcije cilja u prisustvu ograničenja. Nelinearno programiranje je izuzetno važan alat u gotovo svakom području inženjeringa, operacionog istraživanja i matematike, jer se mnogi praktični problemi ne mogu uspešno modelirati kao

linearni [50]. Opšte nelinearno programiranje može se zapisati na sledeći način:

$$max f(x) \tag{9}$$

Uz ograničenja:

$$\begin{aligned} & g_i(x) \leq 0, \quad i = 1, 2, ..., m_1 \\ & h_i(x) = 0, \quad i = m_1 + 1, ..., m(= m_1 + m_2) \\ & x \in X \end{aligned} \tag{10}$$

gde su f, $g_1, g_2, ..., g_{m1}$, $h_{m1+1}, h_{m1+2}, ..., h_m$ stvarne funkcije definisane na E_n, X je podskup E_n, a x je n-dimenzionalni realni vektor sa komponentama $x_1, x_2, ..., x_n$. Gore navedeni problem treba rešiti za vrednosti promenljivih $x_1, x_2, ..., x_n$ koje zadovoljavaju ograničenja, a istovremeno minimizuju funkciju f. Funkcija f se obično naziva *funkcija cilja* (eng. objectiv function), ili *kriterijumska funkcija* (eng. criterion function). Ograničenje $g_i(x) \leq 0$ se naziva *ograničenje nejednakosti* (eng. inequality constraint), a svako od ograničenja $h_i = 0$ naziva se *ograničenje jednakosti* (eng. inequality constraint). Skup X, koji obično može uključivati donje i gornje granice promenljivih, najčešće se naziva *ograničenjem domena* (eng. domain constraint). Vektor $x \in X$ koji zadovoljava sva ograničenja naziva se *prihvatljivim rešenjem problema*. Skup svih takvih rešenja čini *prihvatljivu oblast*. Cilj takvog nelinearnog optimizacionog problema je da pronađe prihvatljivu tačku $\bar{x}$ takvu da važi $f(x) \leq f(\bar{x})$ za svaku prihvatljivu tačku x. Takva tačka se naziva *optimalnim rešenjem*. Za razliku od linearnih optimizacionih problema, konvencionalne metode rešenja za nelinearne optimizacione probleme su veoma složene i nisu vrlo efikasne.

3.1.3 Podela optimizacionih problema na osnovu broja funkcija cilja

Razlikujemo optimizacione probleme sa jednom i sa više funkcija cilja. Optimizacija sa više funkcija cilja (eng. Multi-Objectiv Optimization - MOO) se odnosi na istovremenu optimizaciju (tj. maksimizaciju i/ili minimizaciju) dve ili više funkcije cilja, koje su često u konfliktu jedna s drugom [100]. Ovaj optimizacioni problem

može se formulisati na sledeći način:

$$Optimize\,\langle f_1(x), f_2(x), ..., f_n(x)\rangle \tag{11}$$

Uz ograničenja:

$$\begin{aligned} &g_i(x) \leq 0, \quad i = 1, 2, ..., n_i \\ &h_i(x) = 0, \quad i = 1, 2, ..., n_e \\ &x_l < x < x_u \end{aligned} \tag{12}$$

gde je n broj funkcija cilja koje se istovremeno optimizuju, x je vektor sa m promenljivih odlučivanja (kontinualnih i/ili diskretnih) sa donjom (x_l) i gornjom (x_u) granicom, respektivno, n_i i n_e su brojevi (g) i (h) ograničenja, redom. Prostor izvodljivosti (eng. feasible space), F, je skup vektora x koji zadovoljavaju sva ograničenja i granice u jednačini (12).

Kod MOO je potrebno odrediti skup vrednosti x, koje obezbeđuju najbolja kompromisna rešenja za sve navedene funkcije cilja. Nije izvodljivo pronaći jedinstveno rešenje koje istovremeno optimizuje suprotne ciljeve. Umesto toga, nalazi se skup rešenja sa sledećim karakteristikama: poboljšanje bilo koje od funkcija cilja nije moguće bez pogoršanja jedne ili više drugih funkcija cilja u optimizacionom problemu. Ova optimalna rešenja se nazivaju *Pareto-optimalna rešenja* (po imenu italijanskog ekonomiste Vilfreda Paretoa). Ona pružaju kvantitativne kompromise između zadatih ciljeva .

Vektor $x^* \in F$ je Pareto optimalan ako ne postoji vektor $x \in F$ koji bi poboljšao neku funkciju cilja, bez istovremenog pogoršanja barem jedne druge ciljne funkcije. Pareto-optimalna rešenja se nazivaju i *nedomominantna rešenja*. Ovo znači da, ukoliko imamo dva rešenja $S1$ i $S2$, $S1$ je dominantnije od $S2$ ako je rešenje $S1$ jednako dobaro kao i $S2$ u svim funkcijama cilja i bolje bar u jednom. Ako nijedno od rešenja ne dominira nad drugim, onda su oba nedomominantna jedno u odnosu na drugo.

Pareto-optimalni front (tj. skup nedomominantnih rešenja) je osnovni cilj u MOO. Izbor rešenja sa Pareto-optimalnog fronta zavisi od sklonosti donosioca odluka, znanja o proučavanom problemu, kao i optimalnih vrednosti promenljivih odlučivanja. Iz

tog razloga donosilac odluka, na osnovu svog iskustva i intuicije, treba izabrati najprikladnije rešenje za implementaciju ili dalje istraživanje.

Generalno, dobar Pareto-optimalni front treba da ima dva poželjna svojstva: nedomominantna rešenja su ravnomerno raspoređena i pokrivaju širok spektar vrednosti proučavanih ciljeva. Međutim, pronalaženje takvog Pareto-optimalnog fronta može biti veoma teško, posebno za velike probleme sa neprekidnim i nekonveksnim prostorima pretrage. U MOO, koncept lokalnog minimuma se zamenjuje lokalnim Pareto-optimalnim frontom, čije prisustvo može izazvati probleme u konvergenciji MOO metoda ka globalnom Pareto-optimalnom frontu.

Pareto-optimalni frontovi mogu biti konkavni, konveksni, ili mogu sadržati i konkavne i konveksne delove sa prekidima. Bolja nedomominantna rešenja se dobijaju pomoću MOO metoda za probleme sa konveksnim Pareto frontovima u poređenju sa onima koji imaju konkavne Pareto frontove. Pareto-optimalni frontovi sa prekidima su česti u inženjerskim problemima i zahtevaju složeniju analizu. Dimenzija problema može uticati na veličinu i oblik skupa Pareto-optimalnih rešenja, što određuje performanse MOO metoda. Takođe, kompleksnost i težina rešavanja MOO problema, kao i težina analize njihovih rešenja, verovatno će se povećavati sa brojem ciljeva.

3.2 Klase kompleksnosti

Kompleksnost problema je ekvivalentna složenosti najboljeg algoritma koji rešava taj problem. Problem je rešiv (ili lak) ako postoji polinomsko vreme potrebno za rešavanje tog problema. Problem je nerešiv (ili težak) ako ne postoji algoritam sa polinomskim vremenom koji može da reši taj problem.

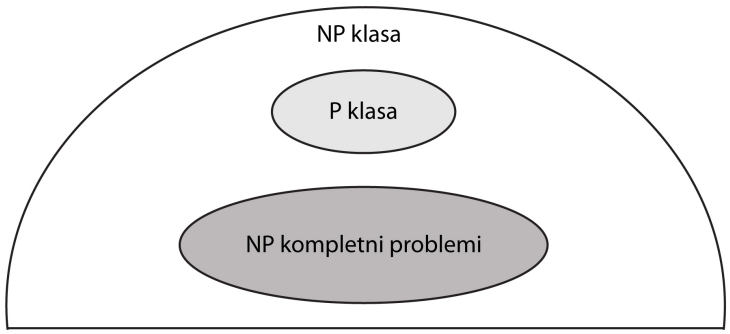
Teorija kompleksnosti problema (eng. complexity theory of problems) se bavi problemima odlučivanja (eng. decision problems). Problem odlučivanja uvek ima odgovor *da* ili *ne*.

Optimizacioni problem uvek može biti redukovan na problem odlučivanja.

Važan aspekt teorije izračunavanja (eng. Computational Theory) je kategorizacija problema u klase kompleksnosti. Klasa kompleksnosti predstavlja skup svih problema koji se mogu rešiti korišćenjem određenih računarskih resursa. Postoje dve važne klase problema: P (eng. polynomial time) i NP (eng. non-deterministic polynomial time).

Klasa složenosti P predstavlja skup svih problema odlučivanja koji se mogu rešiti deterministčkim algoritmom u polinomskom vremenu. Deterministički algoritam je polinomski za problem odlučivanja A, ako je njegova najgora složenost ograničena polinomskom funkcijom $p(n)$, gde n predstavlja veličinu ulazne instance I. Dakle, klasa P predstavlja skup problema za koje postoji poznati algoritam sa polinomskim vremenom, koji može rešiti problem. Problemi koji pripadaju klasi P su relativno *laki* za rešavanje.

NP klasa kompleksnosti predstavlja skup svih problema odlučivanja koji se mogu rešiti pomoću nedeterminističkog algoritma u polinomskom vremenu. Nedeterministički algoritam sadrži jednu, ili više tačaka izbora u kojima je moguće nastaviti izvršavanje na različite načine, bez precizne specifikacije koja će opcija biti odabrana. Koriste se sledeće osnove: *izbor*(eng. choice) koji predlaže rešenje, *proveru* (eng. check) koja u polinomskom vremenu proverava da li predloženo rešenje daje pozitivan ili negativan odgovor, *uspeh* (eng. success) kada algoritam odgovori *da* nakon primene provere, i *neuspeh* (eng. fail) kada algoritam ne odgovori *da*. Ako *izbor* predloži rešenje koje daje odgovor *da* i postoji mogućnost da se to proveri, tada je složenost izračunavanja polinomska.



Slika 7: *Klase kompleksnosti problema odlučivanja*

Pitanje da li je $P = NP$ jedno od najvažnijih otvorenih pitanja zbog širokog uticaja koji bi odgovor imao na teoriju kompleksnosti izračunavanja (eng. computational complexity theory). Očigledno je da za svaki problem u P imamo nedeterministički algoritam koji ga rešava. Dakle, $P \subseteq NP$ (Slika 7). Međutim, pretpostavka da je $P \subset NP$, je i dalje otvoreno pitanje.

Problem odlučivanja $A \in NP$ je NP-kompletan (eng. NP complete) ako su svi ostali problemi iz klase NP polinomski redukovani na problem A. Slika 7 prikazuje odnos između P, NP i NP-kompletnih problema. Ako postoji polinomiski determi-

nistički algoritam koji rešava NP-kompletan problem, tada se svi problemi iz klase NP mogu rešiti u polinomskom vremenu.

NP-teški (eng. NP hard) problemi su optimizacioni problemi čiji su pripadajući problemi odlučivanja NP-kompletni. Većina optimizacionih problema u stvarnom svetu su NP-teški, za koje ne postoje efikasni algoritmi. Za njihovo optimalno rešavanje je potrebno eksponencijalno vreme (osim ako je $P = NP$). Metahruristike predstavljaju važnu alternativu za rešavanje ove klase problema [119].

4 Metode i algoritmi mašinskog učenja

Mašinsko učenje (eng. Machine Learning - ML) se bavi dizajniranjem algoritama koji omogućavaju računarima da uče iz podataka. Učenje ne mora nužno uključivati svest, već je u pitanju pronalaženje statističkih pravilnosti, ili drugih obrazaca u podacima. Zbog toga se mnogi algoritmi mašinskog učenja bitno razlikuju od pristupa koji bi ljudi primenili prilikom učenja [11].

Mašinsko učenje je fenomen povezan sa poljem veštačke inteligencije (eng. Artificial Intelligence – AI). Postoje dve ključne faze formiranja efikasnog ML modela: trening i testiranje. Trening faza je intenzivna faza istraživanja, koja uključuje obezbeđivanje označenih ili neoznačenih ulaza sistemu. Sistem zatim čuva ove trening ulaze u prostoru karakteristika kako bi ih koristio za buduće predviđanje. Konačno, u fazi testiranja, sistemu se daje neoznačeni ulaz za koji treba da predvidi tačan izlaz [6].

Jednostavno rečeno, ML koristi poznate podatke u svom prostoru karakteristika da bi predvideo rezultate za neoznačene podatke. Na taj način uspešan ML model može da se oslanja na svoja iskustva i razumevanja kako bi predvideo rezultate.

Veštačka inteligencija, posebno mašinsko učenje, su se brzo razvili poslednjih godina u oblasti analize podataka i računarstva, što omogućava aplikacijama da funkcionišu na inteligentan način. ML obično pruža sistemima sposobnost da uče i unapređuju se iz iskustva automatski, bez potrebe za posebnim programiranjem, i generalno se smatraju najpopularnijim novim tehnologijama. Algoritmi mašinskog učenja mogu se kategorisati u četiri glavne vrste: nadgledano, nenadgledano, polunadgledano i pojačano učenje.

U oblasti algoritama mašinskog učenja postoje tehnike klasifikacije, regresije, klasterovanja podataka, inženjeringa karakteristika i smanjenja dimenzionalnosti, učenja pravila povezanosti ili tehnike pojačanog učenja koje se koriste za izradu sistema zasnovanih na podacima. Osim toga, tehnika dubokog učenja potiče od veštačkih neuronskih mreža koje se mogu koristiti za inteligentnu analizu podataka, a poznata je kao deo šire porodice pristupa mašinskom učenju. Veoma je bitno odabrati odgovarajući algoritam učenja koji je pogodan za ciljnu aplikaciju u određenom domenu. Razlog tome je što je svrha različitih algoritama učenja različita, čak i rezultat različitih algoritama učenja u sličnoj kategoriji može varirati u zavisnosti

od karakteristika podataka. Iz tog razloga je važno razumeti principe različitih algoritama mašinskog učenja i njihovu primenljivost u različitim realnim područjima primene [106].

4.1 Vrste podataka

Dostupnost podataka se smatra ključnim za izradu modela mašinskog učenja. Podaci mogu biti različitih oblika, kao što su:

- Strukturirani podaci - imaju dobro definisanu strukturu, pridržavaju se modela podataka sledeći standardni redosled, visoko su organizovani i lako dostupni i koriste se od strane entiteta, ili računarskih programa. Strukturirani podaci obično se čuvaju u dobro definisanim šemama, poput relacionih baza podataka, odnosno u tabelarnom formatu. Primeri strukturiranih podataka uključuju imena, datume, adrese, brojeve kreditnih kartica, informacije o akcijama, geolokaciju itd.
- Nestrukturirani podaci - to su podaci koji nemaju predefinisan format ili organizaciju, što ih čini mnogo težim za uočavanje, obradu i analizu. Najčešće sadrže tekstualne i multimedijalne materijale. Primeri nestrukturiranih podataka uključuju senzorske podatke, e-poštu, blogove, dokumente za obradu teksta, PDF fajlove, audio fajlove, video zapise, slike, prezentacije, veb stranice i mnoge druge vrste poslovnih dokumenata.
- Polustrukturirani podaci - ovi podaci nisu smešteni u relacione baze podataka kao što je to slučaj sa strukturiranim podacima, ali imaju određena organizaciona svojstva koja olakšavaju analizu. Primeri polustrukturiranih podataka su: HTML, XML, JSON dokumenti, NoSQL baze podataka itd.
- Metapodaci - nisu uobičajeni oblik podataka, već su to "podaci o podacima".S druge strane, metapodaci opisuju relevantne informacije o podacima i daju im veći značaj za korisnike podataka. Osnovni primer metapodataka može biti autor dokumenta, veličina fajla, datum generisanja dokumenta, ključne reči za definisanje dokumenta, itd [106].

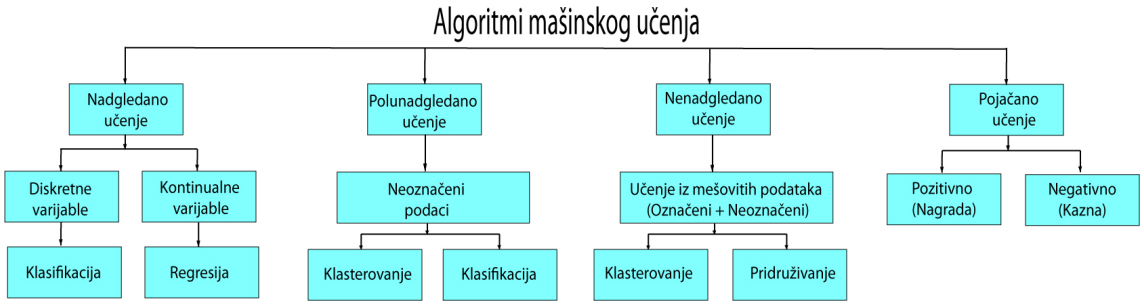
4.2 Pregled modela mašinskog učenja

ML tehnologija ima tri glavne vrste modela:

- nadgledano učenje (eng. Supervised Learning),

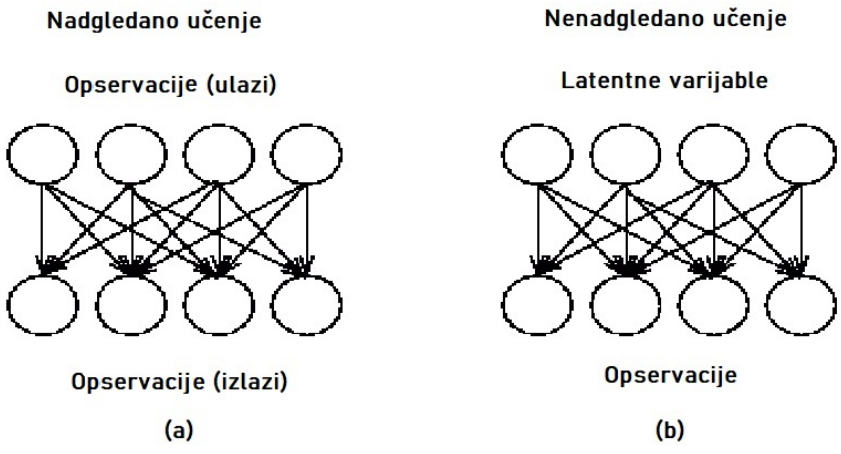
- polunadgledano učenje (eng. Semi-Supervised Learning),
- nenadgledano učenje (eng. Unsupervised Learning) i
- pojačano učenje (eng. Reinforcement Learning - RL).

Svaka vrsta ML modela ima nekoliko zajedničkih algoritama. Na Slici 8 prikazana je podela algoritama mašinskog učenja.



Slika 8: *Podela modela mašinskog učenja*

Nadgledano učenje se najčešće primenjuje u problemima klasifikacije, jer je cilj često naučiti računar da prepozna klasifikacioni sistem koji smo kreirali. Prepoznavanje brojeva je jedan od primera klasifikacionog učenja. Klasifikaciono učenje je pogodno za svaki problem gde je dedukcija klasifikacije korisna i gde je klasifikaciju lako odrediti. U nekim slučajevima možda čak nije ni potrebno unapred odrediti klasifikacije za svaki slučaj problema, ako agent može sam da ih odredi. To bi bio primer nenadgledanog učenja u kontekstu klasifikacije [11]. U nenadgledanom učenju,

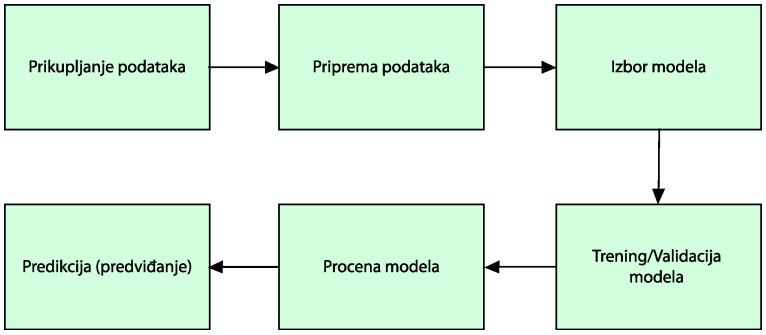


Slika 9: *a) Nadgledano i b) Nenadgledano učenje [11]*

za sva posmatranja se pretpostavlja da su uzrokovana latentnim varijablama. Primeri nadgledanog i nenadgledanog učenja prikazani su na Slici 9.

4.2.1 Nadgledano učenje

Najvažniji model mašinskog učenja je model nadgledanog učenja (Slika 10). On se uglavnom koristi za realne primene. Ovaj model se koristi za predviđanje rezultata na osnovu određenih skupova ulaznih podataka i para ulaznih/izlaznih primera. Svaki trening skup podataka uključuje par ulaznih obeležja, vektora ulaza i željene vrednosti izlaza koje se nazivaju supervizorni signal. Ovi primeri se koriste za treniranje ML algoritama. Cilj algoritama za trening u nadgledanom učenju je da predvide vrednost jednog, ili više rezultata, putem različitih ulaznih karakteristika.



Slika 10: *Blok-šema procesa nadgledanog učenja*

Karakteristična osobina modela nadgledanog učenja je uključenost čoveka. Ljudska uključenost je neophodna na početku kako bi se konstruisao skup podataka, koji kasnije samostalno funkcioniše generalizujući i učeći na osnovu primera koje dobija preko ulaza. Za formiranje skupa podataka, prvo se modelu mašinskog učenja obezbeđuju parovi ulaza i željenih izlaza. Zatim model pronalazi način da samostalno generiše izlaze. Glavni problem se javlja kada model mora samostalno da predvidi izlaz za novi ulaz bez ljudske pomoći. Iz tog razloga veoma je važno obezbediti tačnost predloženog modela.

Nedostatak nadgledanog učenja je to što zahteva veliki broj označenih podataka. Modeli nadgledanog učenja se široko koriste kod problema klasifikacije i regresije.

Klasifikacija, ili predviđanje su osnovni ciljevi upotrebe metoda mašinskog učenja. Ovi metodi klasifikuju i predviđaju klasne oznake koristeći unapred definisanu listu primera. Uzorci klasifikacije pripadaju potpuno određenoj klasi, ili uopšte ne pripadaju ni jednoj klasi. Oni ne pripadaju ni jednoj klasi delimično. Vrednosti koje nedostaju negativno utiču na procese klasifikacije i predviđanja.

Postoje dve forme klasifikacije: binarna i višeklasna. Kod binarne klasifikacije

postoje dva skupa klasa, a ulazni podaci su raspoređeni u ta dva skupa. Primer je predviđanja DA/NE ili klasifikacija e-pošte u dve kategorije, spam i ne-spam. Ove klasifikacione klase se tumače kao 0 i 1. Sa druge strane, višeklasna klasifikacija se odnosi na tri ili više predvidljivih klasa. Primer bi bio identifikacija stadijuma raka. Ovde se klase definišu kao 0, 1, 2 itd [6].

4.2.2 Nenadgledano učenje

Kod nenadgledanog učenja cilj je da računar nauči kako da nešto radi, a da mu pritom čovek ne govori na koji način. Postoji dva pristupa ovom učenju. Prvi pristup je da agentu ne dajemo eksplicitne kategorizacije, već koristimo neku vrstu sistema nagrada da bismo pokazali uspeh. Ovde nije cilj da se proizvede klasifikacija, već da se donesu odluke koje maksimizuju nagrade. Ovaj pristup lepo generalizuje na stvarni svet, gde agenti mogu biti nagrađeni za izvršavanje određenih akcija i kažnjeni za izvršavanje drugih. Često se za nenadgledano učenje može koristiti neka vrsta obučavanja pojačanjem, gde agent zasniva svoje akcije na prethodnim nagradama i kaznama, ne morajući čak ni da uči informacije o tačnom načinu na koji njegove akcije utiču na svet. Na neki način, sve ove informacije nisu potrebne, zato što agent, učeći funkciju nagrade, zna šta treba da radi bez ikakve obrade, jer zna tačnu nagradu koju očekuje da postigne za svaku akciju koju može preduzeti. Ovo može biti izuzetno korisno u slučajevima kada je računanje svake mogućnosti vremenski zahtevno. S druge strane, učenje metodom pokušaja i greške, može biti veoma vremenski zahtevno. Druga vrsta nenadgledanog učenja se naziva grupisanje (klasterovanje). U ovom tipu učenja cilj je da se pronađu sličnosti u trening podacima. Pretpostavka je često da će pronađene grupe prilično dobro odgovarati intuitivnoj klasifikaciji [11].

Jedna od funkcija nenadgledanog mašinskog učenja je identifikacija skrivenih struktura u podacima koje nisu označene. Ovo je postignuto kroz nekoliko uspešnih aplikacija, mada je često teško proceniti njihovu efikasnost. To je zbog nedostatka obuke prilikom korišćenja nenadgledanog mašinskog učenja. Kao rezultat toga, nedostaju indikatori greške, ili nagrade za analizu potencijalnih rešenja. Ovde, signal nagrade služi kao faktor razlike između nadgledanog i nenadgledanog mašinskog učenja. Modeli neuronskih mreža (eng. Neural Network - NN), samoorganizujuća mapa

(eng. Self Organizing Map - SOM) i adaptivna rezonantna teorija (eng. Adaptive Resonance Theory - ART), takođe koriste nenadgledano učenje.

Nenadgledano učenje uključuje transformaciju skupova podataka i klasterovanje. U procesu transformacije, podaci u datasetu se menjaju kako bi bili prikazani u obliku koji je lakši za razumevanje za ljude i algoritme mašinskog učenja. S druge strane, algoritmi klasterovanja razdvajaju skup podataka u značajne grupe povezanih objekata. Algoritam K-srednjih vrednosti (eng. K-means Clustering Algorithm) predstavlja najpoznatiji i najjednostavniji nenadgledani algoritam koji identifikuje klasterovanje sličnih podataka. Ovaj algoritam ima dva koraka: prvi korak je dodeljivanje svake tačke podataka najbližem centru klastera, dok je drugi korak fiksiranje centara klastera kao srednje vrednosti podataka koji su im dodeljeni.

Jedan od glavnih problema u nenadgledanom učenju je procena uspeha. Uspeh nenadgledanog učenja govori da li je algoritam naučio korisne stvari ili ne. Oznake ili izlazi nisu dostupni, pa iz tog razloga pravi izlaz nije poznat. Zbog toga teško odrediti performanse algoritama. Zato se nenadgledano učenje koristi isključivo na eksplorativan način, na primer, za bolje razumevanje podataka. Još jedna važna karakteristika nenadgledanog algoritama je preprocesiranje za nadgledane algoritme [6].

4.2.3 Polunadgledano učenje

Polunadgledano učenje se može definisati kao hibridizacija gore navedenih nadgledanog i nenadgledanog učenja, budući da radi i na označenim i neoznačenim podacima. Zato se nalazi između *učenja bez nadzora* i *učenja sa nadzorom*. U stvarnom svetu, označeni podaci mogu biti retki, dok su neoznačeni podaci brojni, gde je polunadgledano učenje korisno. Krajnji cilj modela polunadgledanog učenja je pružanje boljeg rezultata za predviđanje nego što se dobija samo korišćenjem označenih podatke iz modela. Neke oblasti primene u kojima se koristi polunadgledano učenje uključuju mašinsko prevođenje, otkrivanje prevara, označavanje podataka i klasifikacija teksta [106].

Jedna od grana modela mašinskog učenja je tehnika učenja koja koristi označene i neoznačene podatke za modele mašinskog učenja. U realnom scenariju, neophodno je koristiti manji broj označenih podataka sa velikim brojem neoznačenih podataka

kako bi se postigla veća tačnost učenja. Označavanje skupa podataka zahteva ljudsku uključenost i vreme, što može otežati stvaranje potpuno označenog skupa podataka za obuku i dovesti do visokih troškova. Kao rezultat toga, u određenim slučajevima, polunadgledano učenje može se pokazati kao bolje rešenje. U slučaju ograničenog broja označenih uzoraka, polunadgledano učenje se uglavnom koristi za poboljšanje performansi modela [6].

4.2.4 Pojačano učenje

Pojačano učenje, ili učenje sa pojačanjem, potkrepljeno učenje, i sl. je vrsta algoritma mašinskog učenja koji omogućava softverskim agentima i mašinama da automatski procene optimalno ponašanje u određenom kontekstu ili okruženju, kako bi poboljšali svoju efikasnost. Ova vrsta učenja se zasniva na nagradi ili kazni, a krajnji cilj je koristiti saznanja dobijena iz interakcije sa okruženjem kako bi se preduzele akcije koje povećavaju nagradu ili minimizuju rizik. To je moćan alat za obuku AI modela koji može pomoći u povećanju automatizacije ili optimizaciji operativne efikasnosti složenih sistema poput robotike, autonomne vožnje, proizvodnje i logistike u lancu snabdevanja, međutim, nije poželjno koristiti ga za rešavanje osnovnih ili jednostavnih problema.

Pojačano učenje, zajedno sa nadgledanim i nenadgledanim učenjem, predstavlja jedan od osnovnih modela mašinskog učenja. RL se može koristiti za rešavanje brojnih problema u stvarnom svetu u različitim oblastima, kao što su teorija igara, teorija kontrole, analiza operacija, teorija informacija, optimizacija zasnovana na simulaciji, proizvodnja, logistika lanca snabdevanja, višestruki agenski sistemi, inteligencija roja, upravljanje avionima, upravljanje pokretima robota i mnoge druge [106].

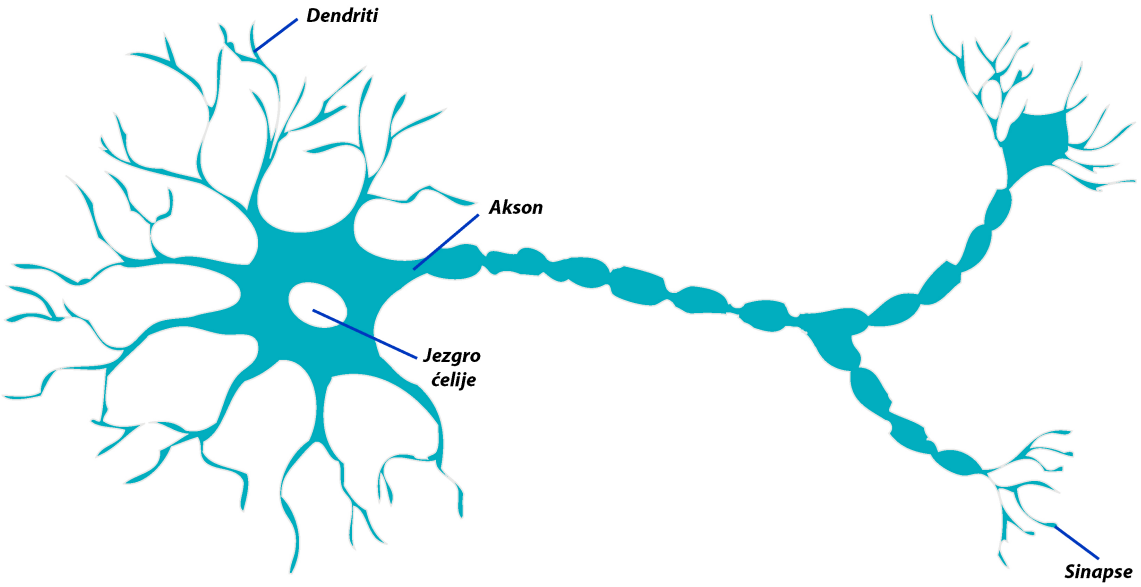
4.3 Neuronske mreže

4.3.1 Biološki i veštački neuron

Veštačke neuronske mreže (eng. Artificial Neural Networks - ANN) predstavljaju tehnike mašinskog učenja koje simuliraju mehanizam učenja u biološkim organizmima. Ljudski nervni sistem sadrži ćelije koje se nazivaju *neuronima*. Neuroni su međusobno povezani pomoću *aksona* i *dendrita*, a regije povezivanja između aksona i dendrita nazivaju se *sinapsama*. Jačina sinaptičkih veza često se menja pod dejstvom

spoljašnjih uticaja. Pod uticajem ovih promena odvija se učenje kod živih organizama. Na Slici 11 prikazan je izgled biološkog neurona.

U veštačkim neuronskim mrežama, simuliran je ovaj biološki mehanizam. Ove mreže se sastoje od računskih jedinica koje se nazivaju neuronima. Računske jedinice povezane su međusobno putem težina, koje imaju istu ulogu kao jačina sinaptičkih veza u biološkim organizmima. Svaki ulaz u neuron je skaliran težinom, koja utiče na funkciju koja se računa u toj jedinici.



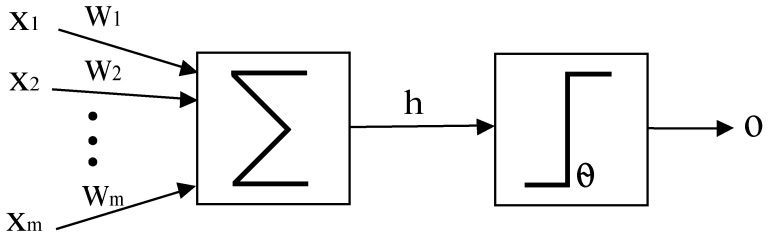
Slika 11: *Biološki neuron*

Proces učenja (treninga) kod veštačkih neuronskih mreža se odvija promenom težina koje povezuju neurone. Baš kao što su spoljni stimulansi potrebni za učenje kod bioloških organizama, kod veštačkih neuronskih mreža ovi stimulansi se obezbeđuju trening podacima koji sadrže primere ulazno-izlaznih parova funkcije koja treba da se nauči. Ovi parovi podataka za obuku unose se u neuronsku mrežu koristeći reprezentacije ulaza kako bi se vršile predikcije izlaza. Podaci za obuku pružaju povratnu informaciju o ispravnosti težina u neuronskoj mreži u zavisnosti od toga koliko dobro predviđeni izlaz, za određeni ulaz, odgovara vrednostima izlaza u podacima za obuku. Greške koje neuronska mreža napravi u računanju funkcije mogu se posmatrati kao vrsta neprijatne povratne informacije u biološkom organizmu, što dovodi do prilagođavanja jačina sinaptičkih veza. Slično tome, težine između neurona se prilagođavaju u neuronskoj mreži kao odgovor na greške u predviđanju. Cilj promene težina je da se modifikuje izračunata funkcija kako bi se predviđanja

učinila tačnijim u budućim iteracijama. Postepenim prilagođavanjem težina između neurona preko mnogih ulazno-izlaznih parova, funkcija koju izračunava neuronska mreža vremenom se usavršava kako bi pružala tačnije predviđanja. Ova sposobnost tačnog izračunavanja funkcija nepoznatih ulaza, uz pomoć obuke sa konačnim skupom ulazno-izlaznih parova, naziva se *generalizacija modela* (eng. model generalization).

Najosnovnije jedinice računanja u neuronskoj mreži inspirisane su tradicionalnim algoritmima mašinskog učenja poput regresije najmanjih kvadrata i logističke regresije. Neuronske mreže dobijaju svoju snagu spajanjem velikog broja osnovnih jedinica i učenjem težina različitih jedinica zajedno, kako bi se minimizovala greška predviđanja. Na ovaj način neuronsku mrežu možemo posmatrati kao računarski graf elementarnih jedinica čija se snaga povećava povezivanjem na određene načine. Kada se neuronska mreža koristi u svojoj najosnovnijoj formi, bez povezivanja više jedinica, algoritmi učenja često se svode na klasične modele mašinskog učenja [4].

Matematički model neurona predstavljen je 1943. godine od strane McCullocha i Pittsa [84]. Ovaj model prikazan je na Slici 12. Svrha matematičkog modela je da se izdvoje samo osnovni elementi potrebni za tačno predstavljanje proučavanog entiteta, uklanjajući sve suvišne detalje.



Slika 12: *Matematički model neurona [82]*

Ulazi x_i se množe težinama w_i, a neuroni sabiraju njihove vrednosti. Ako je zbir veći od praga θ, tada se neuron "pali", odnosno aktivira, dok se u suprotnom ne dešava ništa. Model se sastoji od skupa težinskih ulaza w_i (što odgovara sinapsama), sabirača koji sumira ulazne signale (ekvivalent membrani ćelije koja prikuplja električni napon) i aktivacione funkcije (inicijalno praga) koja odlučuje da li se neuron aktivira za trenutne ulaze. Levo na slici su skupovi ulaznih čvorova (označeni $x_1, x_2, ..., x_m$). Jačina sinapse utiče na jačinu signala, pa množimo ulaz sa težinom sinapse (tako da dobijemo $x_1 \times w_1$ i $x_2 \times w_2$, itd.). Kada svi ovi signali stignu do posmatranog neurona, on ih sabira da vidi da li ima dovoljno snage da se aktivira. To se može

predstaviti kao:

$$h = \sum_{i=1}^{m} w_i x_i \tag{13}$$

McCulloch i Pitts neuron je binarni uređaj koji sabira ulaze (umnožene sa jačinama ili težinama sinapsi) i aktivira se (proizvodi izlaz 1), ili ne aktivira (proizvodi izlaz 0), zavisno od toga da li je ulaz iznad određenog praga. Odluka da li će se aktivirati ili ne (što se naziva funkcija aktivacije), se može napisati kao:

$$o = g(h) = \begin{cases} 1, & ako \quad je \quad h > \theta \\ 0, & ako \quad je \quad h \leq \theta \end{cases} \tag{14}$$

Ovo je veoma jednostavan model, ali iako ovi neuroni mogu izgledati jednostavno, mreža takvih neurona može izvršiti veoma složene računske operacije, pod uslovom da se težine w_i pravilno odrede [82].

4.3.2 Vrste neuronskih mreža

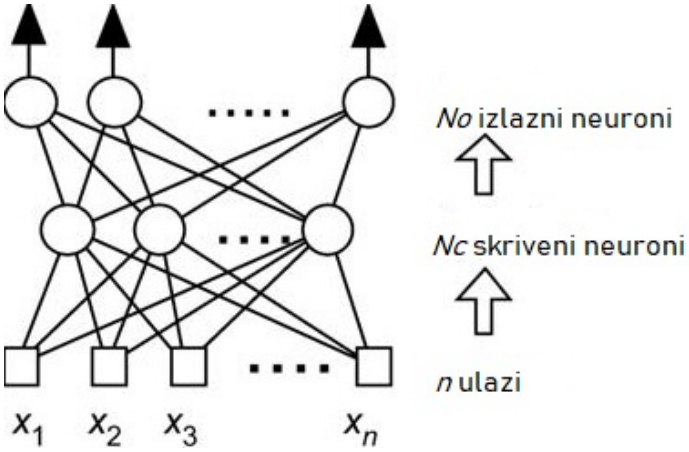
Neuron je nelinearna, parametrizovana funkcija svojih ulaznih promenljivih. Neuronska mreža je kompozicija nelinearnih funkcija dva ili više neurona. Podstoje dve kategorije neuronskih mreža: mreže sa progresijom unapred (eng. feedforward) i rekurentne (eng. feedback) mreže [37].

Neuronske mreže sa progresijom unapred

Neuronska mreža sa progresijom unapred se može predstaviti gtafički skupom neurona koji su međusobno povezani, pri čemu je tok informacija isključivo u smeru unapred (ulaz – izlaz). U grafičkom prikazu, gde neuroni predstavljaju čvorove, a grane veze, graf neuronske mreže je acikličan: nijedna putanja u grafu, prateći veze, ne može se vratiti na početnu tačku.

Neuroni koji izvršavaju konačno izračunavanje, tj. čiji izlazi predstavljaju izlaze mreže, nazivaju se *izlaznim neuronima*; drugi neuroni, koji izvršavaju posredno računanje, nazivaju se *skrivenim neuronima* (Slika 13). Termin *veza* se treba uzeti metaforički. U većini aplikacija, neuroni nisu fizički objekti, tako da veze stvarno ne postoje. Izračunavanja koja vrši svaki neuron implementirana su kao softverski

programi, napisani na bilo kom prikladnom jeziku i izvršavaju se na računaru.

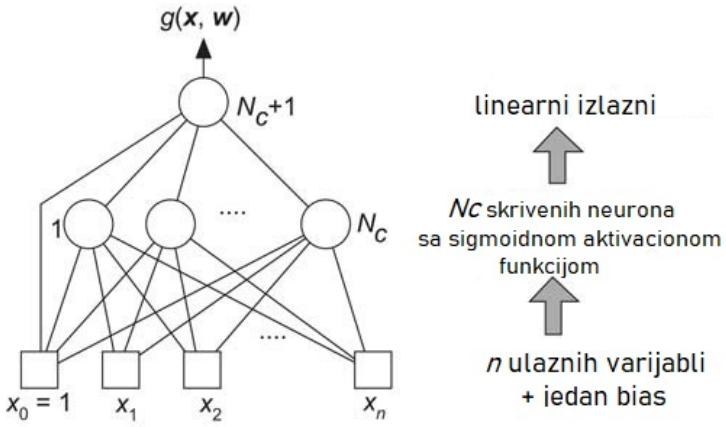


Slika 13: *Primer neuronske mreže [37]*

Višeslojne mreže

Najveću primenu imaju višeslojne neuronske mreže, čiji je primer dat na Slici 13. Prikazana neuronska mreža je sa progresijom unapred. Sastoji se od sa n - ulaza, N_c - skrivenih neurona i N_o - izlaznih neurona. Neuronske mreže sa progresijom unapred su statične, što znači da su izlazi konstantni, ukoliko su ulazi konstantni. Da bi se izračunala funkcija svakog neurona, potrebno je zanemarljivo vreme. Neuronske mreže sa progresijom unapred često se nazivaju *statičnim mrežama*, nasuprot rekurentnim ili *dinamičnim mrežama*.

Neuronske mreže sa progresijom unapred sa više slojeva i sigmoidnim nelinearnostima često se nazivaju višeslojnim perceptronima (eng. Multilayer Perceptron - MLP). U literaturi, često se pominje ulazni sloj i ulazni neuroni kao deo strukture višeslojnog perceptrona. To može biti zbunjujuće jer ulazi (prikazani kao kvadrati na Slici 13, za razliku od neurona koji su prikazani kao krugovi) definitivno nisu neuroni: ne obavljaju nikakvo procesiranje ulaza, već ih samo prosleđuju kao promenljive skrivenim neuronima. Na Slici 14 prikazana je klasa neuronskih mreža sa jednim slojem skrivenih neurona sa sigmoidnom aktivacionom funkcijom i linearnim izlaznim neuronima.



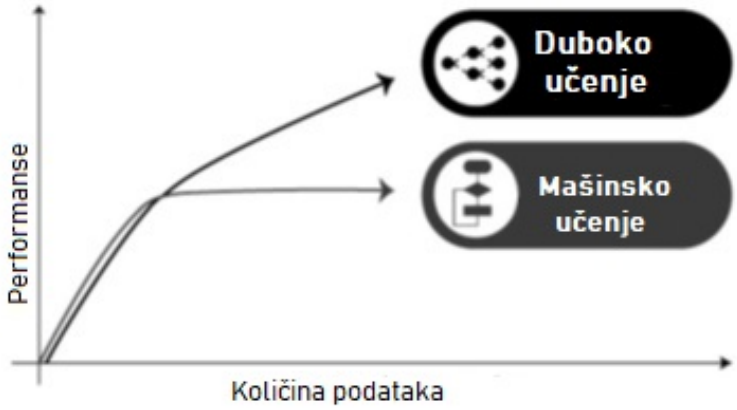
Slika 14: *Neuronska mreža sa n + 1 ulazom, slojem od Nc skrivenih neurona sa sigmoidnom aktivacionom funkcijom, i linearnim izlaznim neuronom [37]*

4.4 Duboko učenje

Arhitekture dubokog učenja (eng. Deep Learning - DL) sastoje se od više slojeva međusobno povezanih čvorova, ili veštačkih neurona, organizovanih na hijerarhijski način. Slojevi uključuju ulazni sloj koji prima podatke, jedan ili veći broj slojeva koji pronalaze relevantne karakteristike (skriveni slojevi) i sloja koji proizvodi željeni rezultat (izlazni sloj). Svaki neuron u sloju prihvata ulaze, primenjuje matematičku operaciju i prosleđuje izlaz sledećem sloju. Kroz proces poznat kao propagacija unazad (eng. backpropagation), mreža prilagođava težine i bias kako bi optimizovala svoje performanse. Glavna prednost dubokog učenja u odnosu na tradicionalne metode mašinskog učenja jesu njegove bolje performanse u mnogim slučajevima, posebno pri učenju iz velikih skupova podataka.

Na Slici 15 su prikazane opšte performanse dubokog učenja u odnosu na mašinsko učenje u zavisnosti od povećanja količine podataka. Međutim, ove performanse mogu varirati u zavisnosti od karakteristika podataka i eksperimentalnog okruženja [106].

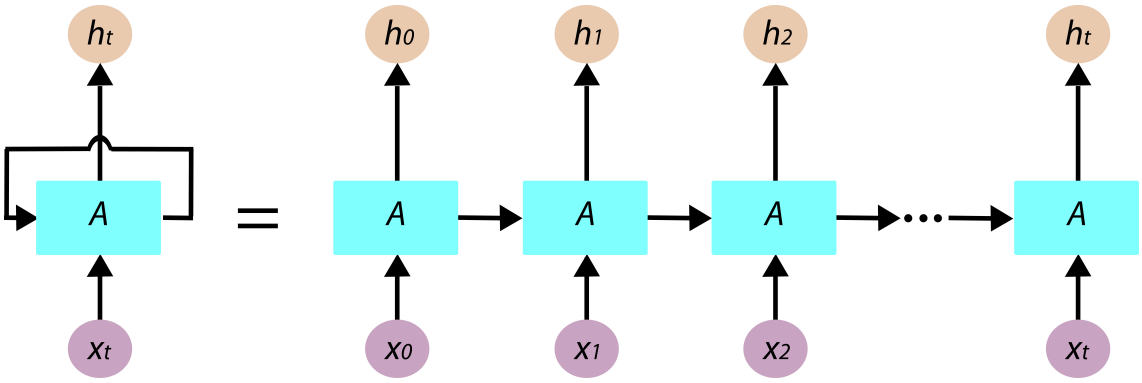
Duboko učenje je tehnologija koja se veoma brzo razvija i koja je donela napredak u mnogim naučnim oblastima kao što su: klasifikacija slika, obrada prirodnog jezika, prepoznavanje govora, itd. Modeli bazirani na dubokom učenju su podskup mašinskog učenja koji imaju za cilj da izvuku karakteristike iz velike količine sirovih podataka. Neke vrste dubokih neuronskih mreža (eng. Deep Neural Network - DNN) su konvolucijske neuronske mreže (eng. Convolutional Neural Network - CNN), rekurentne neuronske mreže (eng. Recurrent Neural Network - RNN), duboke mreže verovanja (eng. Deep Belief Network - DBN), itd.[61].



Slika 15: *Poređenje performansi ML i DL [106]*

4.5 RNN

Rekurentne neuronske mreže predstavljaju arhitekturu mreže specijalizovanu za obradu sekvencijalnih podataka, kao što su rečenice u prirodnom jeziku i vremenske serije. RNN je vrsta veštačke neuronske mreže koja ima sposobnost simulacije vremenskih serija sa dugoročnim strukturnim zavisnostima. Princip rada RNN-a je dodavanje vremenske jedinice kašnjenja i povratnih veza, što omogućava da informacije iz prethodnih stanja budu korišćene u narednim stanjima [19].



Slika 16: *Model rekurentne neuronske mreže [46]*

RNN je klasa nadgledanog modela mašinskog učenja [7]. Sastoji se od slojeva koji su rekurentni, što znači da mreža pamti informacije iz prethodnog izlaza i te informacije mogu biti dodate prilikom izračunavanja trenutnih rezultata. Osnovna arhitektura RNN-a uključuje ulazne jedinice, izlazne jedinice i skrivene jedinice. Skrivene jedinice izvršavaju sve računske operacije prilagođavajući težine kako bi proizvele izlaze [91]. Sposobnost pamćenja informacija iz prethodnih podataka jedna je od najvažnijih karakteristika RNN modela. Model rekurentne mreže prikazan je

na Slici 16 [46].

Jedna RNN ćelija se može predstaviti sledećom jednačinom:

$$h_t = \sigma\left(Wx_t + Uh_{t-1} + b\right) \tag{15}$$

gde su x_t i h_t ulaz i skriveno stanje ćelije u vremenu t, respektivno. W i U su težinska matrica ulaza posmatrane ćelije i težinska matrica rekurentnog ulaza, b je bias neurona, σ je aktivaciona funkcija neurona (sigmoidna funkcija), dok je N broj neurona u posmatranom RNN sloju [7].

Kod RNN mreža mogu se javiti problemi nestajanja (eng. vanishing gradients),ili eksplozije gradijenta (eng. exploding gradients). To znači da će kada je vremenski interval dug, rezultati ranijeg učenja nestati. Kako bi prevazišli ovaj problem, Hochreiter i Schmidhuber su predložili model duge kratkoročne memorije (eng. Long Short-Term Memory - LSTM) [55].

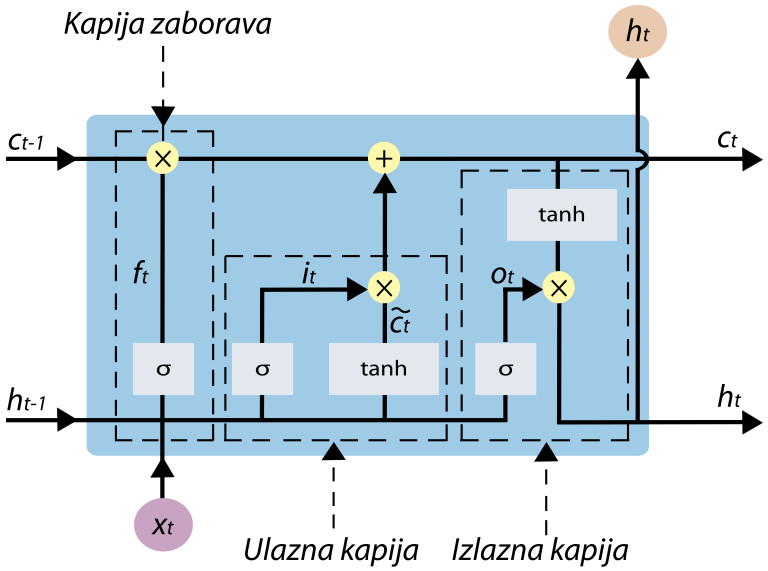
4.5.1 LSTM

Ova vrsta rekurentne neuronske mreže može da nauči dugoročno zavisne informacije [77]. Ona može da pamti odnose između trenutnih informacija i dugoročnih informacija u vremenskim serijama. Skriveni sloj tradicionalne RNN je zamenjen memorijskom ćelijom koja je prikazana na Slici 17 [77, 125].

Osnovna jedinica LSTM-a je memorijski blok koji se uglavnom sastoji od memorijske ćelije i tri upravljačke jedinice: *kapije zaborava* (eng. forget gate), *ulazne kapije* (eng. input gate) i *izlazne kapije* (eng. output gate). Sve upravljačke jedinice imaju istu strukturu i sastoje se od: sigmoidne aktivacione funkcije i operacije množenja. Sigmoidna funkcija je nelinearna aktivaciona funkcija čiji je izlaz u opsegu [0, 1] i definiše stepen prolaska informacija. Izlaz aktivacione funkcije tangens hiperbolički (tanh) je u opsegu [-1, 1] [122].

Ulaz LSTM ćelije se sastoji od prethodnog izlaza h_{t-1} i trenutnog ulaza x_t. Kapija zaborava kontroliše koji elementi vektora stanja ćelije C_{t-1} će biti odbačeni, kako je definisano jednačinom (16) [46].

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{16}$$



Slika 17: *LSTM ćelija*

gde je f_t is izlazni vektor čije su vrednosti u opsegu [0,1], σ je sigmoidna funkcija, W_f i U_f su težinske matrice, a b_f je bias vektor.

Ulazna kapija ažurira informacije koristeći rezultat sigmoidnog sloja i_t prema jednačini:

$$i_t = \sigma\left(W_i x_t + U_i h_{t-1} + b_i\right) \tag{17}$$

gde su W_i, U_i težinske matrice. a b_i je bias vektor.

Nove potencijalne vrednosti vektora stanja ćelije $\tilde{C}_t$ generišu se pomoću funkcije *tanh* i određene su jednačinom (18).

$$\tilde{C}_t = tanh\left(W_c x_t + U_c h_{t-1} + b_c\right) \tag{18}$$

Vrednost C_t se dobija množenjem starog stanja ćelije C_{t-1} sa f_t (da bi se zaboravile nepoželjne informacije) i dodavanjem nove potencijalne informacije $i_t \otimes \tilde{C}_t$, kao što je prikazano u jednačini (19).

$$C_t = f_t \otimes C_{t-1} + i_t \otimes \tilde{C}_t \tag{19}$$

gde simbol $\otimes$ označava množenje elemenata [125, 42].

Vrednost izlazne kapije o_t se izračunava prema sledećoj jednačini:

$$o_t = \sigma\left(W_o x_t + U_o h_{t-1} + b_o\right) \tag{20}$$

gde su W_o, U_o težinske matrice, dok je b_o bias vektor.

Vrednost izlaza h_t se izračunava kao što je dato u jednačini (21).

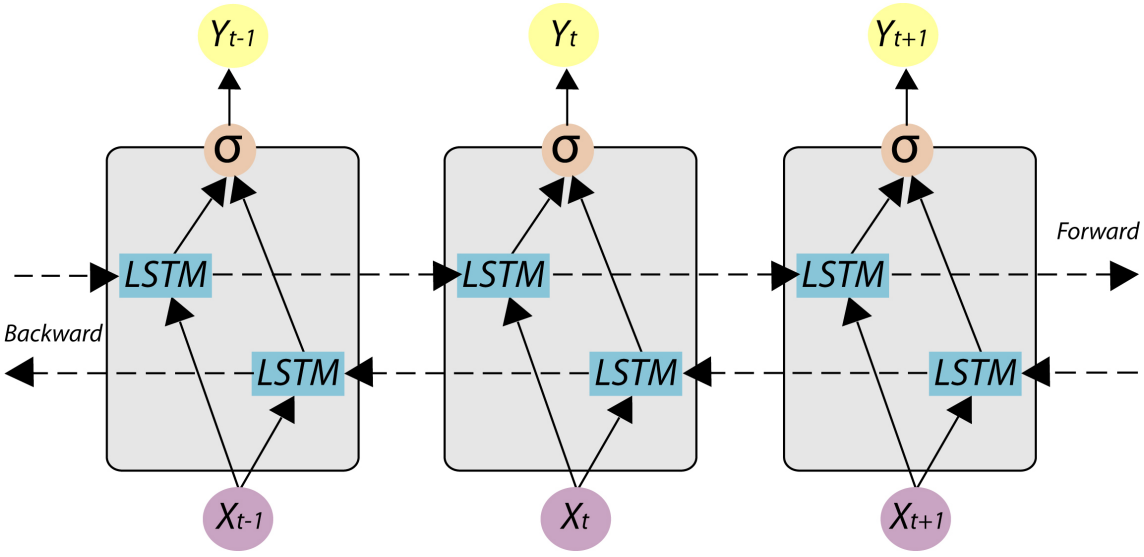
$$h_t = o_t \otimes tanh(C_t) \tag{21}$$

Pomoću funkcije $tanh$, stanje ćelije C_t je skalirano u opsegu [-1, 1], a množenjem sa izlazom izlazne kapije o_t, izračunata je nova izlazna vrednost h_t.

LSTM se može trenirati pomoću istorijskih vremenskih serija. Nakon treninga LSTM mreža se koristi za predviđanje novih podataka.

4.5.2 BiLSTM

Bidirekcionalna LSTM (eng. Bidirectional Long Short-Term Memory - BiLSTM) neuronska mreža se sastoji od dva LSTM modula. Da bi se povećala tačnost predikcije, BiLSTM mreža može koristiti prošla i buduća stanja. Ova rekurentna neuronska mreža, sa propagacijom unapred i unazad, prevazilazi nedostatak informacija o podacima koji je prisutan kod LSTM neuronskih mreža [105]. BiLSTM rešava problem nestajućih i eksplodirajućih gradijenata [129]. Arhitektura BiLSTM mreže prikazana je na Slici 18 [124].



Slika 18: *Arhitektura BiLSTM mreže*

Isti podaci se unose u *forward* i *backward* sloj LSTM mreže. U *forward* sloju LSTM mreže, izračunavanje se vrši unapred od vremena 1 do vremena t. U *backward* sloju LSTM mreže, izračunavanje se vrši unazad od vremena t do vremena 1 [136].

U smeru unapred, informacije o prošlim vrednostima vremenskog niza čuvaju se u $H_{f(t)}$, kako je prikazano u jednačini (22) [124].

$$H_{f(t)} = \psi(W_{fh}X_t + W_{fhh}H_{f(t-1)} + b_{fb}) \tag{22}$$

U smeru unazad, informacije o budućim vrednostima se čuvaju u $H_{b(t)}$, kao što je prikazano u jednačini (23).

$$H_{b(t)} = \psi(W_{bh}X_t + W_{bhh}H_{b(t+1)} + b_b) \tag{23}$$

gde su W_{fh}, W_{fhh}, W_{bh} i W_{bhh} težinske matrice, b_{fb} i b_b predstavljaju bias signale u smerovima unapred i unazad, respektivno. ψ je aktivaciona funkcija rekurentnog sloja.

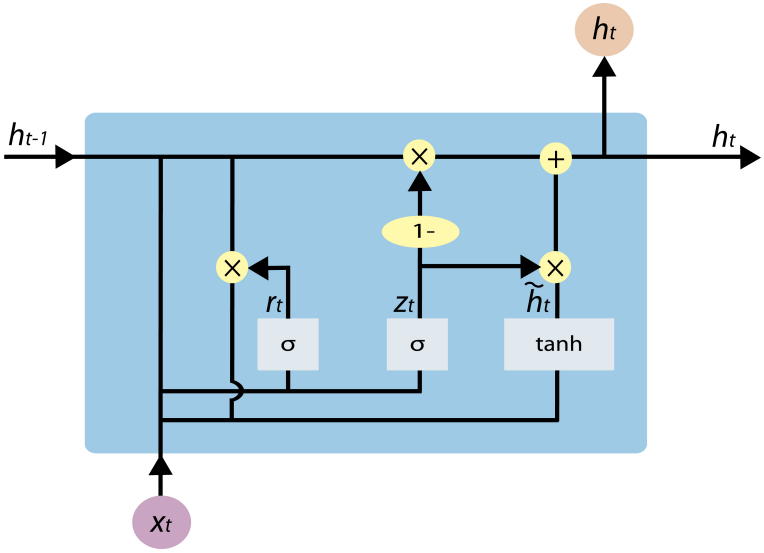
Izlazni vektor BiLSTM-a Y_t je predstavljen jednačinom(24):

$$Y_t = \sigma(W_{fhy}H_{f(t)} + W_{bhy}H_{b(t)} + b_y) \tag{24}$$

gde je W_{fhy} težina između unutrašnjih jedinica i izlaza za smer unapred, W_{bhy} težina između unutrašnjih jedinica i izlaza za smer unazad, σ je aktivaciona funkcija izlaznog sloja, dok b_y predstavlja vektor pomeraja (bias) izlaznog sloja.

4.5.3 GRU

Rekurentna jedinica ograničena kapijama (eng. Gated Recurrent Unit - GRU) Princip GRU je predstavio Cho [28] 2014. godine, kako bi se prevazišao problem nestajuih gradijenata sa kojim se susreću konvencionalne rekurentne mreže i predložila arhitektura sa manje parametara u odnosu na LSTM arhitekturu. Slično kao i LSTM jedinica, GRU koristi kapije koje modulišu protok informacija unutar jedinice. Da bi se smanjio broj parametara, GRU ćelije *ulaznu kapiju* i *kapiju zaborava* integrišu u jednu *kapiju ažuriranja* (eng. update gate). Pored ove kapije, postoji i *kapija reseta* (eng. reset gate). Kapija ažuriranja i kapija reseta određuju, redom, koje informacija treba zadržati, a koje treba zaboraviti [7, 113]. Izgled GRU ćelije prikazan je na Slici 19.



Slika 19: *GRU ćelija*

Način funkcionisanja GRU ćelije prikazan je pomoću sledećih jednačina:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{25}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{26}$$

$$\tilde{h}_t = \tanh(W_h x_t + U_h r_t \otimes h_{t-1} + b_h) \tag{27}$$

$$h_t = (1 - z_t) \otimes h_{t-1} + z_t \otimes \tilde{h}_t \tag{28}$$

gde x_t predstavlja ulaz mreže u trenutku t, z_t i r_t su vektori kapija koji predstavljaju izlaz kapije ažuriranja i kapije resetovanja u trenutku t, $\tilde{h}_t$ i h_t su vektori informacija koji predstavljaju privremeni izlaz i izlaz skrivenog sloja u trenutku t , W_z, U_z, W_r, U_r, W_h i U_h su težinske matrice, b_z, b_r i b_h su bias vektori. $\sigma(x)$ i $tanh(x)$ predstavljaju sigmoidnu i tangens hiperboličku aktivacionu funkciju, respektivno [62].

4.6 Optimizacija hiperparametara

Većina algoritama dubokog učenja dolazi sa velikim brojem hiperparametara, koji kontrolišu različite aspekte ponašanja algoritma. Neki od njih utiču na vreme i trošak memorije potrebne za izvođenje algoritma. Drugi hiperparametri utiču na kvalitet modela koji se dobija tokom procesa obuke i njegovu sposobnost ispravnog

zaključivanja kada se primenjuje na novim ulazima.

Razlikuju se dva pristupa za odabir hiperparametara: manuelni odabir i automatski odabir. Kod manuelnog odabira neophodno je razumevanje uloge hiperparametara i kako modeli mašinskog učenja postižu dobru generalizaciju. Algoritmi automatskog odabira hiperparametara značajno smanjuju potrebu za razumevanjem tih ideja. Međutim, ovi algoritmi su znatno zahtevniji sa računarske tačke gledišta [52].

Dok se parametri za obuku modela mašinskog učenja prilagođavaju tokom faze obuke, vrednosti hiperparametara (ili meta-parametara) moraju biti navedene pre faze učenja. Cilj je pronaći skup vrednosti hiperparametara koji daje najbolji model za podatke u razumnom vremenskom okviru.

Većina modela korišćenih u mašinskom učenju ima skup hiperparametara koji utiču na to kako algoritam za učenje prilagođava model postojećim podacima. Ove hiperparametre treba razlikovati od internih parametara modela, koje se adaptivno prilagođavaju za rešavanje problema. Na primer, hiperparametri neuronskih mreža obično određuju arhitekturu mreže (broj i tip slojeva, broj i tip čvorova, itd.).

Cilj optimizacije hiperparametara je pronaći hiperparametre datog modela koji daju najbolje performanse, koje su merene na validacionom skupu. Ovo se može prikazati sledećom jednačinom [8]:

$$x^* = arg\min_{x\epsilon\mathbb{N}} f(x) \tag{29}$$

gde $f(x)$ predstavlja funkciju cilja koju treba minimizovati (kao što su RMSE, ili stopa greške) evaluiranu na validacionom skupu; x^* je skup hiperparametara koji daje najmanju vrednost rezultata, a x može uzeti bilo koju vrednost u domenu $\mathbb{N}$. Jednačina (29) se detaljnije može zapisati kao:

$$x^* = arg\min_{x\epsilon\mathbb{N}} f(x, \gamma^*; S_{validation}) \tag{30}$$

Jednačina (30) uključuje unutrašnju optimizaciju koja se koristi za pronalaženje γ^*, optimalne vrednosti γ, za trenutnu vrednost x:

$$\gamma^* = arg\min_{\gamma\epsilon\Gamma} f(x, \gamma; S_{train}) \tag{31}$$

$S_{validation}$ i S_{train} označavaju validacioni i trening skup podataka, respektivno,

dok γ predstavlja skup naučenih parametara u domenu Γ.

Proces validacije je znatno složeniji nego što je ovde opisano. Svaka kombinacija vrednosti hiperparametara može kao rezultat dati drugačiji model, a jednačina (29) procenjuje i upoređuje funkciju cilja za različite modele. Proces pronalaženja najboljeg modela, iz skupa modela koji su generisani različitim postavkama hiperparametara, naziva se *selekcija modela* (eng. model selection). Da bi procenili očekivane rezultate preko nepoznate raspodele, potrebno je izvršiti podelu skupa podataka na skupove za treniranje, validaciju i testiranje. Postojeći algoritmi za optimizaciju hiperparametara dubokog učenja zahtevaju značajne računarske resurse

Postoje dva uzroka ove neefikasnosti: jedan je povezan sa prostorom pretrage, koji može biti diskretan, a diskretna optimizacija je NP-teška. Drugi uzrok je što je procena funkcije cilja za pronalaženje ocene skupa resursno zahtevna: svaki put kada proveravamo različite vrednosti hiperparametara, moramo trenirati model na skupu podataka za trening, zatim vršiti predviđanja na validacionom skupu podataka i, na kraju, izračunati metriku validacije. Ova optimizacija se obično sprovodi ponovnim treniranjem više modela sa različitim kombinacijama vrednosti hiperparametara i procenom njihovih performansi. Ponovno treniranje i evaluacija za jedan skup vrednosti hiperparametara naziva se *pokušaj* (eng. trail).

Izbor dobre optimizacije je težak zadatak, a problem smanjenja prostora pretrage i vremena treniranja je veoma važno. Da bi se ovo postiglo, može se izvršiti smanjenje broja hiperparametara i karakteristika, smanjenjem skupa za treniranje i korišćenjem dodatnih funkcija cilja. Sa brojem hiperparametara, broj pokušaja se povećava eksponencijalno. Zbog toga je moguće smanjiti broj hiperparametara. Hiperparametri se mogu rangirati i izabrati na osnovu funkcionalne analize varijanse funkcije cilja. Ovaj metod se naziva *analiza osetljivosti* (eng. sensitivity analysis).

Smanjenje broja pokušaja nije dovoljno ako je vreme treniranja za svaki pokušaj dugo, pa bi u tom slučaju trebalo smanjiti skup za treniranje. Korišćenje dodatnih funkcija cilja moguće je redukovati prostor pretrage za optimalnu konfiguraciju hiperparametara [8].

Popularnost algoritama, kao što su logističke regresije imetod potpornih vektora (eng. Support Vector Machine - SVM), proizlazi delom iz njihove sposobnosti da dobro rade sa samo jednim, ili dva optimizovana hiperparametra. Neuronske mreže

mogu ponekad dobro raditi sa samo nekoliko optimizovanih hiperparametara, ali često je potrebno optimizovati znatno veći broj hiperparametara. Manuelno podešavanje hiperparametara može vrlo dobro funkcionisati kada korisnik ima saznanja od strane drugih koji su radili na istom tipu aplikacije i arhitekture, ili kada korisnik ima dosta vremena za određivanje vrednosti hiperparametara. Ukoliko ovo nije slučaj, za određivanje vrednosti hiperparametara mogu se koristiti automatizovani algoritmi.

4.6.1 Metodi za optimizaciju hiperparametara

Najjednostavniji način za optimizaciju hiperparametara je *gramzivi* (eng. greedy) pristup, kod koga se vrši istraživanje lokalne okoline datog skupa hiperparametara - promena jednog hiperparametra u jednom trenutku i merenje promene performansi. Jedina informacija dobijena ovom analizom je kako različite vrednosti hiperparametara funkcionišu sa jednom konfiguracijom ostalih hiperparametara, tako da se ovim pristupom ne mogu dobiti dobri rezultati.

Iz tog razloga se koriste sistematičniji pristupi. Ovde će biti opisani neki od njih, i to: pretraga mreže, slučajna pretraga, Nelder-Mead i Bajesova optimizacija.

Metoda pretrage mreže

Najčešće korišćena tehnika optimizacije hiperparametara kombinuje pretragu mreže (eng. Grid Search - GS) i manuelno podešavanje (eng. manual tuning). Postoji nekoliko razloga zbog čega se ovo radi:

- Manuelna optimizacija daje istraživačima određeni uvid u proces optimizacije.
- Ne postoji tehnička prepreka za manuelnu optimizaciju.
- GS je jednostavna za implementaciju, a paralelizacija je trivijalna.
- GS obično pronalazi bolje rešenje od čisto manuelne sekvencijalne optimizacije (u istom vremenskom periodu).
- GS je pouzdana u prostorima niskih dimenzija (na primer, 1D, 2D).

Međutim, GS ima problem dimenzionalnosti, jer broj zajedničkih vrednosti eksponencijalno raste sa brojem hiperparametara. Iz tog razloga, se GS ne preporučuje za optimizaciju velikog broja hiperparametara [8].

Metoda slučajnog pretraživanja

Metoda slučajnog pretraživanja (eng. Random Search - RS) sastoji se u uzimanju uzoraka iz prostora parametara, sledeći određenu raspodelu za svaki od parametara. Koristeći isti broj iteracija, RS obično daje bolje rezultate od GS ili složenijih metoda optimizacije hiperparametara. Posebno u prostorima većih dimenzija, računarski resursi potrebni za RS metode su značajno niži nego za GS. RS najbolje funkcioniše pod pretpostavkom da svi hiperparametri nisu podjednako bitni.

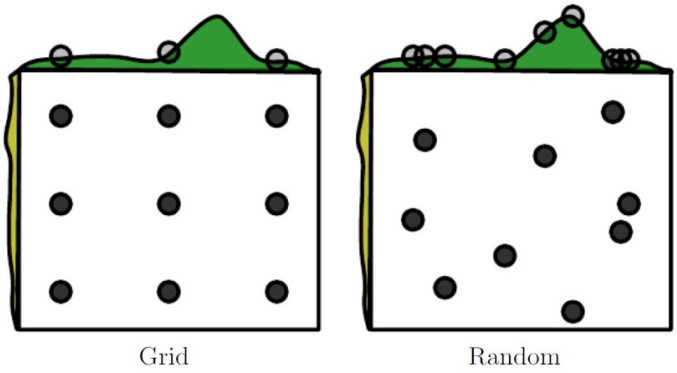
Ostale prednosti RS su:

- Eksperiment se može zaustaviti u bilo kom trenutku. Ključno je definisati dobar kriterijum za zaustavljanje koji predstavlja kompromis između tačnosti i vremena izračunavanja.
- Eksperimentu se mogu dodati nove iteracije, bez potrebe za prilagođavanjem mreže i obaveznog izvođenja mnogo većeg eksperimenta.
- Svaka iteracija može se izvršiti asinhrono, pa su iz tog razloga RS metode relativno jednostavne za implementaciju na paralelnim računarskim arhitekturama.
- Ako računar koji izvodi iteraciju ne uspe, iz bilo kog razloga, njegova iteracija može biti, ili napuštena, ili ponovo pokrenuta, bez ugrožavanja optimizacije.

Na Slici 20 je ilustrovano poređenje GS i RS algoritama za optimizaciju hiperparametara. Pikazana su samo dva hiperparametra, iako ih u praksi obično ima mnogo više. Slika ilustruje tipičan slučaj gde samo neki hiperparametri imaju značajan uticaj na rezultat. Na ovoj ilustraciji, samo hiperparametar na horizontalnoj osi ima značajan efekat. Grid pretraga troši količinu računarskih resursa koja je eksponencijalna u broju nebitnih hiperparametara, dok random pretraga testira jedinstvenu vrednost svakog bitnog hiperparametra na gotovo svakom pokušaju [52].

Nelder-Mead optimaizacija

Kod optimizacije hiperparametara često se suočavamo sa izazovima kao što su: funkcije koje nisu diferencijabilne, višestruki ciljevi, velika dimenzionalnost, mešovite promenljive, višestruki lokalni minimumi, diskretni prostor pretrage, itd. Optimizacija bez gradijenata se odnosi na rešavanje problema optimizacije koristeći algoritme koji ne zahtevaju informacije o gradijentima, već samo vrednosti funkcije cilja. Za razliku od metoda zasnovanih na gradijentima u konveksnom prostoru pretrage, metode bez



Slika 20: *Poređenje GS i RS algoritama za optimizaciju hiperparametara [52]*

korišćenja gradijenta nemaju garanciju da će pronaći globalni optimum. U metode optimizacije bez korišćenja gradijenta spadaju: Nelder-Mead (NMA), Simulirano kaljenje (eng. Simulated Annealing - SA), Evolutivni algoritmi (eng. Evolutionary Algorithm - EA) i Optimizacija rojem čestica (eng. Particle Swarm Optimization - PSO).

NMA je predstavljen 1965. godine i vrši pretragu u n-dimenzionalnom prostoru koristeći heurističke ideje. Metoda koristi koncept simpleksa, strukturu u n-dimenzionalnom prostoru koju čine n + 1 tačaka koje nisu koplanarne. NMA održava skup od n + 1 test tačaka raspoređenih kao simpleks, a zatim modifikuje simpleks pri svakoj iteraciji koristeći četiri operacije: refleksiju, ekspanziju, kontrakciju i sužavanje. Svaka od ovih operacija generiše novu tačku. Niz operacija koje se izvršavaju u jednoj iteraciji zavisi od vrednosti funkcije cilja na novoj tački u odnosu na ostale tačke. Predviđa se ponašanje funkcije cilja na svakoj test tački, kako bi se pronašla nova test tačka i zamenila jedna od starih test tačaka itd. Ako je ova tačka bolja od trenutno najbolje tačke, pokušava se kretanje duž ove linije. Ako nova tačka nije mnogo bolja od prethodne vrednosti, onda se simpleks sužava ka boljoj tački. Iako je davno nastala, ova tehnika je i dalje veoma popularna [52].

NMA je korišćen u optimizaciji konvolutivnih neuronskih mreža, u kombinaciji sa relativno malim skupom podataka za optimizaciju. Dobar je za funkcije cilja koje su glatke, unimodalne i nemaju previše šuma. Nedostatak ove metode je što ne daje dobre rezultate za probleme sa više (od otprilike 10) promenljivih. Iznad ovog broja promenljivih, konvergencija postaje sve teža.

Bajesova optimizacija

RS i GS ne uzimaju u obzir prethodne rezultate i nastavljaju pretragu celokupnog prostora pretrage. Za razliku od njih, Bajesova optimizacija (eng. Bayesian optimization) iterativno računa posteriornu (uslovnu) distribuciju funkcija koja najbolje opisuje funkciju cilja. Kako broj posmatranja raste, posteriorna distribucija se poboljšava, a algoritam postaje sigurniji u vezi sa oblastima u prostoru hiperparametara koje vredi istraživati, a koje ne. Bajesova optimizacija prati prethodne rezultate evaluacija koje koristi kako bi formirao probabilistički model koji mapira hiperparametre na verovatnoću vrednosti funkcije cilja. Ovaj model se naziva *zamena* (eng. surogate) za funkciju cilja. Probabilistički model zamene se iterativno ažurira nakon svake evaluacije funkcije cilja [52].

5 Taksonomija optimizacionih algoritama

Podelu optimizacionih algoritama moguće je izvršiti na više načina. Jedan je podela na determinističke i stohaističke algoritme. Takođe, moguće je izvršiti podelu na egzaktne i aproksimativne algoritme.

Većina klasičnih, ili konvencionalnih algoritama, spada u grupu *determinističkih algoritama*. Na primer, simplex metoda u linearnom programiranju je deterministička. Neki deterministički optimizacioni algoritmi koriste informacije o gradijentu i spadaju u algoritme bazirane na gradijentu. Poznati Newton-Raphson algoritam spada u ovu kategoriju, jer koristi vrednosti funkcija i njihove izvode, i veoma se dobro pokazao za smooth unimodalne probleme (problemi koji imaju glatku funkciju cilja i jednostruki lokalni minimum ili maksimum). Problem nastaje ukoliko postoji prekid u funkciji cilja. U tom slučaju je bolje koristiti algoritam koji nije baziran na gradijentu. Algoritmi koji nisu bazirani na gradijentu, odnosno algoritmi bez gradijenta, ne koriste nikakve izvode, već samo vrednosti funkcija. Hooke-Jeeves algoritam pretrage obrasca (eng. Hooke-Jeeves Pattern Search Algorithm) i Nelder-Mead metod (eng. Nelder-Mead method) su primeri algoritama bez gradijenta.

Stohastičke algoritme možemo svrstati u dve kategorije: heurističke i metaheurističke. Heuristika znači 'pronaći', ili 'otkriti metodom pokušaja i grešaka'. Kvalitetna rešenja za teške optimizacione probleme mogu se pronaći u razumnom vremenskom okviru, ali nema garancije da će se dostići optimalna rešenja. Ovi algoritmi uglavnom rade dobro i mogu se primenjivati kada nije neophodno da se dobiju najbolja rešenja, već dobra rešenja koja su lako dostižna.

Heuristika je metod putem kojeg se, metodom pokušaja i grešaka, generišu prihvatljiva rešenja za složene probleme u razumnom vremenskom okviru. Složenost problema čini nemogućim pretraživanje svakog mogućeg rešenja ili kombinacije, pa je cilj pronaći dobro izvodljivo rešenje u prihvatljivom vremenskom okviru. Ideja je imati efikasan, ali praktičan algoritam koji će uglavnom moći da generiše rešenja visokog kvaliteta. Među pronađenim rešenjima visokog kvaliteta, može se očekivati da će neka biti gotovo optimalna, mada nema garancije za takvu optimalnost

Heuristike spadaju u aproksimativne algoritme i mogu se svrstati u dve kategorije: specifične heuristike i metaheuristike [119].

Metaheuristički algoritmi predstavljaju unapređene heuristike, jer *meta* znači

'iznad', ili 'na višem nivou' i generalno se bolje izvode od običnih heuristika. Svi metaheuristički algoritmi koriste određenu kompromisnu kombinaciju randomizacije (slučajnosti) i lokalne pretrage. Treba napomenuti, da ne postoji usaglašena definicija heuristika i metaheuristika u literaturi, jer neki autori koriste 'heuristike' i 'metaheuristike' naizmenično. U novije vreme trend je da se svi stohastički algoritmi koji uključuju randomizaciju i lokalnu pretragu nazivaju metaheuristikama. Randomizacija omogućava da se izađe iz lokalne pretrage i pređe na globalno pretraživanje. Iz tog razloga skoro svi metaheuristički algoritmi teže da budu pogodni za globalnu optimizaciju.

Dve glavne komponente svakog metaheurističkog algoritma su: intenzifikacija (eksploatacija) i diverzifikacija (eksploracija). *Diverzifikacija* znači generisanje raznovrsnih rešenja u cilju istraživanja prostora pretrage na globalnom nivou. Sa druge strane, *intenzifikacija* predstavlja fokusiranje pretrage u lokalnoj oblasti eksploatacijom informacije da je u toj oblasti pronađeno trenutno dobro rešenje. Selekcija najboljih rešenja obezbeđuje konvergenciju rešenja ka optimalnosti, dok diverzifikacija putem randomizacije izbegava zadržavanje rešenja u lokalnim optimumima i istovremeno povećava raznolikost rešenja. Dobra kombinacija ovih komponenti može osigurati dostizanje globalne optimalnosti [131].

5.1 Kategorizacija metaheurističkih algoritama

Stohastički algoritmi, uključujući metaheuristike, često su neophodni u računarskoj nauci kako bi se nosili sa NP-teškim izazovima, jer deterministički algoritmi nisu praktični za rešavanje ove vrste problema [94]. Metaheuristički algoritmi mogu se klasifikovati u kategorije koje oponašaju prirodne procese kako bi vodili proces pretrage. Na primer, neki metodi su inspirisani evolucijom, prirodnim odabirom ili kolektivnim ponašanjem ptica i insekata [116, 41, 43].

Najpoznatije grupe pristupa metaheuristikama uključuju algoritme inspirisane prirodom, koji obuhvataju genetske algoritme i inteligenciju roja, kao i algoritme zasnovane na određenim fizičkim fenomenima (npr. oluja, gravitaciona i elektromagnetna polja). Drugi pristupi uključuju one koji oponašaju aspekte ljudskog ponašanja poput učenja, razmišljanja ili aktivnosti na društvenim mrežama, kao i one koji proizlaze iz osnovnih matematičkih zakona koji usmeravaju pretragu, na primer,

korišćenjem oscilacija trigonometrijskih funkcija.

Metodi inteligencije roja bazirani su na kooperativnim akcijama grupa relativno jednostavnih jedinki, poput rojeva insekata ili jata ptica, koje mogu pokazivati iznenađujuće koordinisane i sofisticirane obrasce ponašanja dok obavljaju osnovne zadatke preživljavanja poput lova, traženja hrane, parenja ili migracije [21, 1]. Ovi metodi su pokazali značajan potencijal u rešavanju stvarnih NP-teških problema. Međutim, ponekad su poznati po neuspehu. Neki od popularnih algoritama inteligencije roja uključuju optimizaciju rojem čestica (eng. Particle Swarm Optimization - PSO) [70], optimizaciju kolonijom mrava (eng. Ant Colony Optimization - ACO) [35], algoritam svitaca (eng. Firefly algorithm - FA) [130] i algoritam slepog miša (eng. Bat Algorithm - BA) [132, 133]. U poslednjih nekoliko godina, razvijena je visoko efikasna grupa metaheuristika baziranih na matematičkim funkcijama i njihovim obrascima ponašanja kako bi usmeravale proces pretrage, pri čemu su najznačajniji primeri algoritam sinus-kosinus (eng. Sine-cosine algorithm - SCA) [89] i algoritam aritmetičke optimizacije (eng. The Arithmetic Optimization Algorithm - AOA) [3].

NFL teorema (eng. No Free Lunch - NFL) je glavni razlog za postojanje tolike raznolikosti optimizacionih metodologija. NFL tvrdi da ne postoji jedan algoritam koji može univerzalno dominirati u svakom zadatku optimizacije. Iz tog razloga, dok jedan algoritam može dobro raditi na određenom problemu, može potpuno zakazati na drugom, naglašavajući potrebu za raznovrsnim metaheurističkim metodama i potrebu za izborom najprikladnije metode za svaki pojedinačni zadatak optimizacije.

Algoritmi zasnovani na populaciji nedavno su postali uobičajeni izbor za rešavanje različitih stvarnih problema. Ovi algoritmi su korisni u mnogim oblastima, poput predviđanja slučajeva COVID-19 [139, 141], organizacije računarskih usluga na zahtev [12, 25, 23, 140], optimizacije bežičnih senzora i IoT-a [138, 142], selekcije karakteristika [22, 16], obrade i klasifikacije medicinskih slika [24, 137], rešavanja globalnih optimizacionih problema [118, 97], identifikacije prevare sa kreditnim karticama [63, 96], monitoring i predviđanje zagađenja vazduha [13, 66], detekcije napada u mrežama i računarskim sistemima [18, 115], predviđanja proizvodnje energije i opterećenja energijom [15, 117], i optimizacije različitih modela mašinskog učenja [104, 88, 47, 17, 14, 65, 26, 64].

U literaturi se može naći više različitih podela metaheurističkih algoritama. Neke

od njih su na [119]:

- *Prirodom inspirisane metaheuristike* i *metaheuristike koje nisu inspirisane prirodom*

Mnoge metaheuristike su inspirisane prirodnim procesima. Tako su evolutivni algoritmi i veštački imunološki sistemi inspirisani biologijom, kolonije mrava i pčela i optimizacija rojem čestica inspirisana inteligencijom rojeva različitih vrsta, dok je simulirano kaljenje inspirisano fizikom.

- *Metaheuristike koje su bazirane na korišćenju memorije* i na *metaheuristike koje koriste metoda bez memorije*

Neki metaheuristički algoritmi ne koriste memorije, tj. ne koriste informacije koje su dobijene tokom pretrage. Predstavnici ove klase su: lokalna pretraga, GRASP (eng. Greedy Randomized Adaptive Search Procedure) i simulirano kaljenje. Kod drugih metaheuristika koristi se memorija u kojoj su smeštene informacije dobijene tokom pretrage, npr. kratkoročna i dugoročna memorija u tabu pretrazi.

- *Determinističke*, naspram *stohastičkih metaheuristika*

Primer determinističkih metaheuristika su lokalna pretraga i tabu pretraga. U stohastičkim metaheuristikama se tokom pretrage primenjuju neka slučajna pravila (npr. simulirano kaljenje, evolutivni algoritmi). U determinističkim algoritmima, korišćenje istog početnog rešenja vodiće do istog krajnjeg rešenja, dok u stohastičkim metaheuristikama iz istog početnog rešenja mogu biti dobijena različita krajnja rešenja. Prilikom procene performansi metaheurističkih algoritama ova karakteristika se mora uzeti u obzir.

- *Pretraga zasnovana na populaciji*, naspram *pretrage zasnovane na jednom rešenju*

Algoritmi zasnovani na jednom rešenju (npr. lokalna pretraga, simulirano kaljenje) manipulišu i transformišu jedno rešenje tokom pretrage, dok se kod algoritama zasnovanih na populaciji (npr. inteligencija roja, evolutivni algoritmi) dobija cela populacija rešenja. Ove dve porodice metaheuristika se međusobno dopunjuju: metaheuristike zasnovane na jednom rešenju su orijentisane na eksploataciju; imaju moć da intenziviraju pretragu u lokalnim regionima. Metaheuristike zasnovane na populaciji su orijentisane na eksploraciju, omogućavaju bolju diverzifikaciju u celom prostoru pretrage.

- *Iterativne*, naspram *gramzivih metaheuristika*

U iterativnim algoritmima, počinje se sa kompletnim rešenjem (ili populacijom rešenja), koje se transformiše pri svakoj iteraciji korišćenjem određenih operatora pretrage. Gramzivi algoritmi počinju od praznog rešenja, i pri svakom koraku se dodeljuje vrednost promenljive problema, sve dok se ne dobije potpuno rešenje. Većina metaheuristika spada u grupu iterativnih algoritama.

Za potrebe ove disertacije, razmatera se podela na metaheuristike koje nisu inspirisane prirodom i one koje su inspirisane prirodom.

5.2 Metaheuristike koje nisu inspirisane prirodom

U ovu grupu metaheuristika, kao što i sam naziv kaže, spadaju oni algoritmi koji nisu inspirisani prirodnim procesima, niti ponašanjem životinja u prirodi. Postoji veliki broj metaheuristika koje pripadaju ovoj kategoriji, a samo neke od njih, će biti opsane u ovoj disertaciji. To su Monte Karlo metod (eng. Monte Carlo method), lokalna pretraga, gramzivi algoritmi i tabu pretraga.

5.2.1 Monte Karlo metod

Razvoj metode Monte Karlo započeo je 1946. godine od strane matematičara i fizičara Stanisława Ulama, koji je bio uključen u Manhattan projekt, koji je imao za cilj da razvije prvu atomsku bombu. Ovu ideju je podelio sa Johnom Von Neumannom, a 1949. godine u Journal of the American Statistical Association, objavljen je članak u kome su ova dva naučnika predstavila *Metodu Monte Karlo* [87]. Naziv ove tehnike za statističke simulacije predložio je Metropolis, inspirisan interesovanjem koji je Ulam imao za igru pokera [49].

Metode Monte Karla su kolekcija različitih metoda koje u osnovi izvode isti proces. Ovaj proces uključuje izvođenje mnogo simulacija koristeći slučajne brojeve i verovatnoću kako bi se dobila aproksimacija odgovora na problem. Osnovna karakteristika metoda Monte Karla je upotreba slučajnih brojeva u simulacijama. Zapravo, ove metode dobijaju svoje kolektivno ime po tome što Monte Karlo, glavni grad Monaka, ima mnogo kazina, a rulet kazina dobar je primer generatora slučajnih brojeva. Tek sa razvojem računarske tehnologije, ova tehnika je postala šire korišćena. To je zato što računari sada mogu izvesti milione simulacija mnogo efikasnije i brže nego ranije.

Ovo je važan faktor jer znači da tehnika može brzo pružiti približan odgovor i sa višim nivoom tačnosti, jer što se više simulacija izvrši, to će aproksimacija biti tačnija. Treba napomenuti da metode pružaju samo približno rešenje. Iz tog razloga se analiza *greške aproksimacije* može smatrati ključnim faktorom kada se vrši procena ovih metoda. Različite metode mogu imati različite nivoe tačnosti, mada to često može zavisiti od određenih okolnosti, pa nivo tačnosti nekih metoda varira u zavisnosti od problema [95].

Kod Monte Karlo simulacije nastoji se da se prati 'vremenska zavisnost' modela čije promene ili rast ne napreduju po nekom strogo unapred definisanom obrascu, već na stohastički način koji zavisi od sekvence slučajnih brojeva generisanih tokom simulacije. Sa drugom, različitom sekvencom slučajnih brojeva, simulacija neće dati identične rezultate, ali će pružiti vrednosti koje se slažu sa onima dobijenim iz prve sekvence unutar neke statističke greške [73].

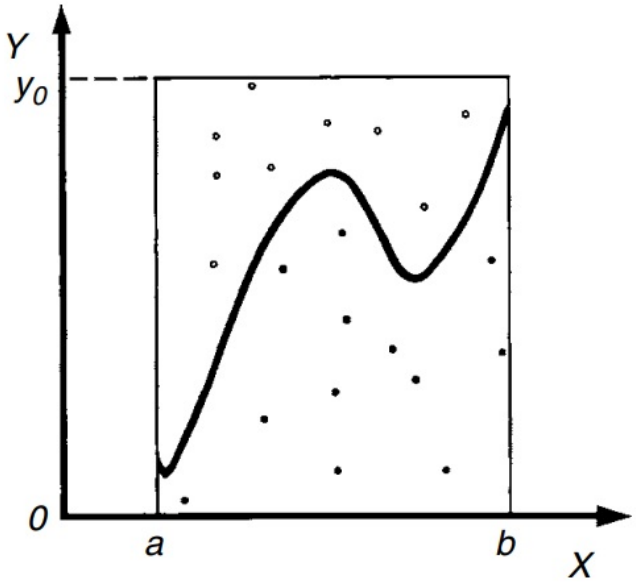
Jedna od najjednostavnijih i najefikasnijih primena metoda Monte Karla je evaluacija određenih integrala koji su nerazrešivi analitičkim tehnikama. Ovde je opisan metod primenjen na jednodimenzionalne integrale, ali ove tehnike se lako mogu proširiti i često su najefikasnije kada se primenjuju na višedimenzionalne integrale. U najjednostavnijem slučaju želimo dobiti integral funkcije $f(x)$ preko nekog fiksnog intervala:

$$y = \int_a^b f(x)dx \tag{32}$$

Jednostavna reprezentacija 'pogodi ili promaši' Monte Karlo integracije funkcije f(x), prikazane neprekidnom krivom, između $x = a$ i $x = b$. N tačaka se nasumično ubacuje u oblast, N_0 od njih pada ispod krive. Integral se procenjuje korišćenjem jednačine (33) [73]

Na Slici 21 prikazana je vizuelna reprezentacija ovog problema. Jednostavno Monte Karlo rešenje ovog problema putem 'pogodi ili promaši' (ili prihvatanje-odbijanje) metode je da se nacrta oblast koja se prostire od a do b i od 0 do y_0 gde je $y_0 > f(x)$ kroz ceo ovaj interval. Koristeći slučajne brojeve izvučene iz uniformne raspodele, nasumično ubacujemo N tačaka u oblast i biramo N_0, koji pada ispod $f(x)$ za svaku vrednost x. Procena integrala se dobija na sledeći način:

$$y_{est} = (N_0/N) \times [y_0(b-a)] \tag{33}$$



Slika 21: *Grafička reprezentacija integracije funkcije f(x) metodom Monte Karlo[73]*

Ova procena postaje sve preciznija kako $N \to \infty$ i konačno će konvergirati ka tačnom odgovoru. Ova tehnika je primer 'jednostavnog uzorkovanja' Monte Karlo metode i očigledno zavisi od kvaliteta sekvence slučajnih brojeva koja se koristi. Nezavisne procene mogu se dobiti primenom istog pristupa sa različitim sekvencama slučajnih brojeva, a poređenjem ovih vrednosti može se utvrditi preciznost postupka.

Još jedan tip metode Monte Karla naziva se *jednostavna metoda* (eng. crude method). U ovom pristupu biramo nasumično N vrednosti x , a zatim određujemo $f(x)$ za svaku vrednost, tako da se procena integrala dobija pomoću formule:

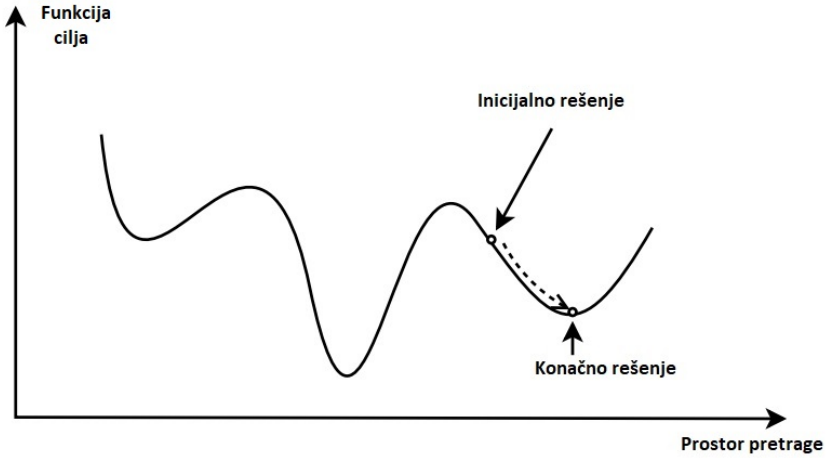
$$y_{est} = \frac{1}{N} \sum_i f(x_i) \tag{34}$$

gde sa povećanjem broja izabranih vrednosti x, procenjeni odgovor konačno konvergira ka tačnom rezultatu. U jednostavnoj varijanti ovog metoda, interval se može podeliti u set nejednakih podintervala i izvršiti odvojena Monte Karlo integracija za svaki podinterval. Na mestima gde je funkcija velika, uzorkovanje može biti obimno, a manje truda može se uložiti u one podintervale gde je funkcija mala [73]

5.2.2 Lokalna pretraga

Može se smatrati da je lokalna pretraga najstarija i najjednostavnija metaheuristička metoda. Ona počinje zadavanjem inicijalnog rešenja. Pri svakoj iteraciji,

heuristika zamenjuje trenutno rešenje susedom koji poboljšava funkciju cilja. Pretraga se zaustavlja kada su svi potencijalni susedi gori od trenutnog rešenja, što znači da je dostignut lokalni optimum. Za veliki broj suseda, potencijalna rešenja mogu biti podskup susedstva. Glavni cilj ograničavanja susedstva je ubrzanje pretrage. Varijante lokalne pretrage mogu se razlikovati u zavisnosti od redosleda generisanja susednih rešenja (deterministički/stohastički) i strategije izbora (izbor susednog rešenja) (Slika 22).



Slika 22: *Lokalna pretraga*

Lokalnu pretragu možemo posmatrati kao spuštanje niz graf koji predstavlja prostor pretrage, a može se definisati sa $G = (S, V)$, gde je S skup svih izvodljivih rešenja prostora pretrage, a V vezu između suseda. U grafu G, grana (i, j) će povezivati bilo koja susedna rešenja s_i i s_j. Za dato rešenje s, broj povezanih grana biće $|N(s)|$ (broj suseda).

Algoritam 1 predstavlja algoritam lokalne pretrage [119].

Algorithm 1 LS algoritam

$s = s_0$ Generisanje inicijalnog rešenja s_0;
while nije zadovoljen kriterijum zaustavljanja **do**
  Generisanje suseda (N(s));
  if nema boljeg suseda **then**
    Stop;
  else if then
    $s = s'$ Selaektovanje boljeg suseda $s' \in N(s)$;
  end if
end while
Pronađeno je konačno rešenje (lokalni optimum).

Od početnog rešenja s_0, algoritam će generisati niz $s_1, s_2, ..., s_k$ rešenja sa sledećim karakteristikama:

- Veličina niza k nije unapred poznata
- $s_{i+1} \in N(s_i), \forall i \in [0, k-1]$.
- $f(s_i + 1) < f(s_i), \forall i \in [0, k-1]$
- s_k je lokalni optimum: $f(s_k) < f(s), \forall s \in N(s_k)$.

Pored definisanja početnog rešenja i susedstva, projektovanje algoritama lokalne pretrage mora da se bavi strategijom selekcije suseda koji će odredi sledeće aktuelno rešenje. Kod velikog broja suseda, ovaj metod može imati dugo vreme izvršavanja. Lokalna pretraga dobro radi ukoliko se u prostoru pretrage ne nalazi veliki broj lokalnih optimuma, ili ako je kvalitet različitih lokalnih optimuma manje-više sličan. Međutim, ako je funkcija cilja visoko multimodalna, što je često slučaj kod većine optimizacijskih problema, lokalna pretraga obično nije efikasan metod [119].

5.2.3 Gramzivi algoritmi

U gramzivim, ili konstruktivnim algoritmima, počinje se od nule (prazno rešenje) i konstruiše rešenje dodeljivanjem vrednosti jednoj promenljivoj odlučivanja u svakom trenutku, sve dok se ne generiše potpuno rešenje. U optimizacionom problemu, gde se rešenje može definisati prisustvom/odsustvom konačnog skupa elemenata $E = e_1, e_2, ..., e_n$, funkcija cilja se može definisati kao $f : 2E \rightarrow R$, a prostor pretrage se definiše kao $F \subset 2^E$. Delimično rešenje s može se posmatrati kao podskup $e_1, e_2, ..., e_n$ elemenata e_i iz skupa svih elemenata E. Skup koji definiše početno rešenje je prazan. U svakom koraku, lokalna heuristika se koristi za odabir novog elementa koji će biti uključen u skup. Kada se element e_i jednom odabere da bude deo rešenja, nikada se ne zamenjuje drugim elementom. Ne postoji povratno praćenje već preuzetih odluka. Gramzive heuristike su obično deterministički algoritmi. Ovi algoritmi su popularni, zbog toga što su jednostavni za dizajn i manje složeni u poređenju s iterativnim algoritmima [119].

5.2.4 Tabu pretraga

Algoritam Tabu pretrage (eng. Tabu Search - TS) predložio je Glover [51] 1986. godine. Ovaj algoritam pretrage je veoma popularan za rešavanje problema optimiza-

cije na približan način. Korišćenje memorije koja čuva informacije o procesu pretrage, predstavlja posebnu karakteristiku Tabu pretrage [119].

Kod algoritma Tabu pretrage cela susedstva se istražuju deterministički, dok se u simuliranom kaljenju bira nasumičan sused. Kao i kod lokalne pretrage, kada se pronađe bolji sused, on zamenjuje trenutno rešenje, a kada se dostigne lokalni optimum, pretraga se nastavlja biranjem kandidata koji je gori od trenutnog rešenja. Najbolje rešenje u susedstvu bira se kao novo trenutno rešenje čak i ako ne poboljšava trenutno rešenje. Algoritam Tabu pretrage može se posmatrati kao dinamička transformacija susedstva. Ovo može da dovede do generisanja ciklusa, tako da prethodno posećena rešenja mogu ponovo biti izabrana. Kako bi izbegao cikluse, algoritam Tabu pretrage odbacuje susede koji su već ranije posećeni. On pamti nedavnu putanju pretrage. Ovaj algoritam upravlja memorijom rešenja ili poteza koji su nedavno primenjeni, što se naziva *tabu lista* (eng. tabu list). Ova lista zabranjenih poteza formira kratkoročnu memoriju. Pri svakoj iteraciji TS algoritma, ažurira se kratkoročna memorija. Čuvanje svih posećenih rešenja zahteva vreme i prostor, jer se u toku svake iteracije mora proveriti da li generisano rešenje ne pripada listi svih posećenih rešenja. Tabu lista obično sadrži konstantan broj zabranjenih poteza. Uobičajeno je da se atributi poteza čuvaju u tabu listi.

Uvođenjem pojma *karakteristika rešenja* ili *karakteristika poteza* u tabu listu, mogu se izgubiti neke informacije o memoriji pretrage. Mogu se odbaciti neka rešenja koja još uvek nisu generisana. Tabu lista zabranjenih poteza može biti previše restriktivna, tako da rešenje unapred može biti zabranjeno. Međutim, pod određenim uslovima, nazvanim *kriterijumi aspiracije* (eng. aspiration criteria), zabranjena rešenja mogu biti prihvaćena. Dopuštena susedna rešenja su ona koja nisu zabranjena ili ispunjavaju kriterijume aspiracije.

5.3 Metaheuristike inspirisane prirodom

U grupu metaheuristika inspirisanih prirodom, spadaju oni algoritmi koji su inspirisani nekim prirodnim fenomenom, ili ponanašanjem ljudi i životinja. Ovi algoritmi se mogu svrstati u dve grupe i to: Evolutivni algoritme i metaheuristike inspirisane inteligencijom rojeva. Postoji veliki broj algoritama inspirisanih prirodom, a ovde će biti opisani samo neki od njih i to: Evolutivni algoritm, Genetski algoritm

i Simulirano kaljenje, dok će celo naredno poglavlje biti posvećeno metaheuristikama koje su inspirisane inteligencijom rojeva.

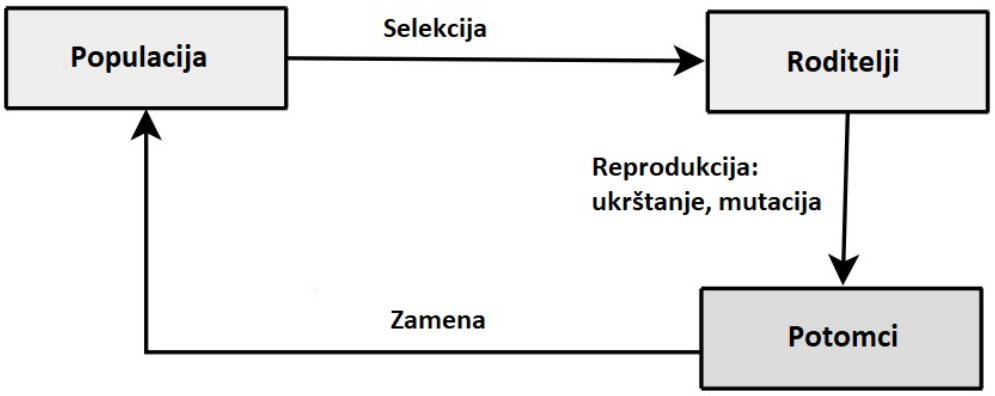
5.3.1 Evolutivni algoritmi

Evolutivni algoritmi (eng. Evolutionary Algorithms - EA) su stohastičke P-metaheuristike (eng. Population-Based), metaheuristike bazirane na populaciji, koje su uspešno primenjivane na mnoge stvarne i kompleksne probleme. Njihov uspeh u rešavanju teških optimizacionih problema u različitim oblastima (kontinualna ili kombinatorna optimizacija, modeliranje i identifikacija sistema, planiranje i kontrola, projektovanje inženjeringa, itd) promovisao je oblast poznatu kao *evolutivno računarstvo* (eng. Evolutionary Computation - EC).

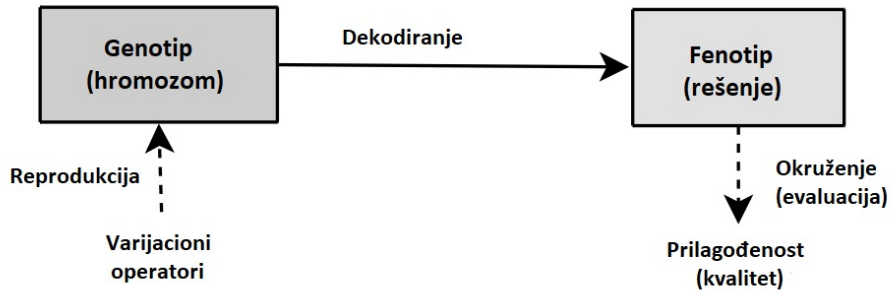
EA su zasnovani na ideji takmičenja. Oni predstavljaju klasu iterativnih optimizacionih algoritama koji simuliraju evoluciju vrsta. Baziraju se na evoluciji populacije jedinki. Na početku se populacija obično generiše nasumično. Svaka jedinka u populaciji predstavlja kodiranu verziju preliminarnog rešenja (eng. tentative solution). Funkcija cilja povezuje vrednost fitnesa sa svakom jedinkom, što ukazuje na njenu prikladnost za problem. U svakom koraku, jedinke se biraju tako da formiraju roditelje, gde je veća verovatnoća da jedinke sa boljim fitnesom budu izabrane. Zatim se odabrane jedinke reprodukuju koristeći operatore varijacije (npr. ukrštanje, mutacija) da bi se generisala nova potomstva. Na kraju, primenjuje se šema zamene da bi se odredilo koje jedinke iz populacije će preživeti. Ova iteracija predstavlja *generaciju*, što je prikazano na Slici 23. Ovaj proces se ponavlja dok se ne ispune kriterijumi zaustavljanja.

U evolutivnim algoritmima, *genotip* predstavlja kodiranje, dok *fenotip* predstavlja rešenje. Dakle, genotip mora biti dekodiran kako bi se generisao fenotip. Operatori varijacije deluju na nivou genotipa, dok će fitnes funkcija koristiti fenotip povezanog pojedinca, kao što je prikazano na Slici 24. Fitnes (prilagođenost) jedinke predstavlja sposobnost te jedinke da preživi u svom okruženju. U slučaju direktnog kodiranja, genotip je sličan fenotipu. U suprotnom (tj. ako se koristi indirektno kodiranje), genotip i fenotip su različite strukture. Tada se koristi funkcija dekodiranja za transformisanje genotipa u fenotip [119].

Evolutivni algoritmi se mogu svrstati u četiri kategorije [114]:



Slika 23: *Generacija u EA*



Slika 24: *Genotip i fenotip u EA*

- Genetski algoritmi (eng. Genetic Algorithms - GA),
- Evolutivno programiranje (eng. Evolutionary Programming),
- Evolutivne strategije (eng. Evolutionary Strategies),
- Genetsko programiranje (eng. Genetic Programming).

5.3.2 Genetski algoritmi

Genetski algoritmi su jedan od najznačajnijih predstavnika EA. GA je netradicionalna tehnika pretrage. Netradicionalne tehnike mogu izbeći nedostatak tradicionalnih tehnika, kao što je pretpostavka o kontinualnim promenljivima. GA je idealno prilagođen za rešavanje problema sa diskretnim promenljivama i u većini slučajeva nalazi globalno optimalno rešenje [92].

GA predstavljaju simulacije prirodne selekcije, koje mogu rešiti probleme optimizacije. Osnovne karakteristike prirodne selekcije su:

- Biološki sistem uključuje populaciju jedinki, od kojih mnoge imaju sposobnost

reprodukcije.

- Jedinke imaju ograničen vek.
- Postoji varijacija u populaciji.
- Sposobnost preživljavanja je u pozitivnoj korelaciji sa sposobnošću reprodukcije.

Svaka od pojedinačnih karakteristika, ili osobina, kontrolisana je parom gena. Ljudska genetika se iz tog razloga naziva *diploidna*, što znači da geni za svaku osobinu dolaze u paru. Neke biljke i životinje su *haploidne*, što znači da svaka osobina zavisi od jednog skupa gena. Drugi organizmi su *poliploidni*, što znači da svaka osobina zavisi od više od dva skupa gena. U diploidnoj genetici, neki geni su dominantni, dok su drugi recesivni. Ako se u jedinki pojave i dominantni i recesivni gen, tada će dominantni gen odrediti osobinu koja će se pojaviti kod te jedinke. Jedini način da recesivni gen odredi osobinu je ako su oba gena ista [114].

Geni se ne prenose nepromenjeni od roditelja na potomstvo, već se menjaju. Ovaj proces promene gena, naziva se *mutacija*. Mutacije su uzrokovane nesavršenošću života i bioloških procesa, uključujući radijaciju i bolesti. Većina mutacija je neutralna, tako da ima malo, ili nimalo uticaja na potomstvo. Većina mutacija koje utiču na biološko potomstvo je štetno. Postoji više od 6.000 mutacija pojedinačnih gena koje izazivaju bolesti, i one se javljaju kod jednog od svakih 200 potomaka. Neke genetske poremećaje možemo primetiti odmah po rođenju, dok se drugi mogu ispoljiti tek kasnije u životu.

Postoje i mutacije koje su korisne. Mutacija koja je praćena prirodnom selekcijom, pomaže poboljšanju preživljavanja vrste. Mutacija je generalno štetna za pojedince, ali može biti korisna za vrstu kao celinu.

Binarni genetski algoritam

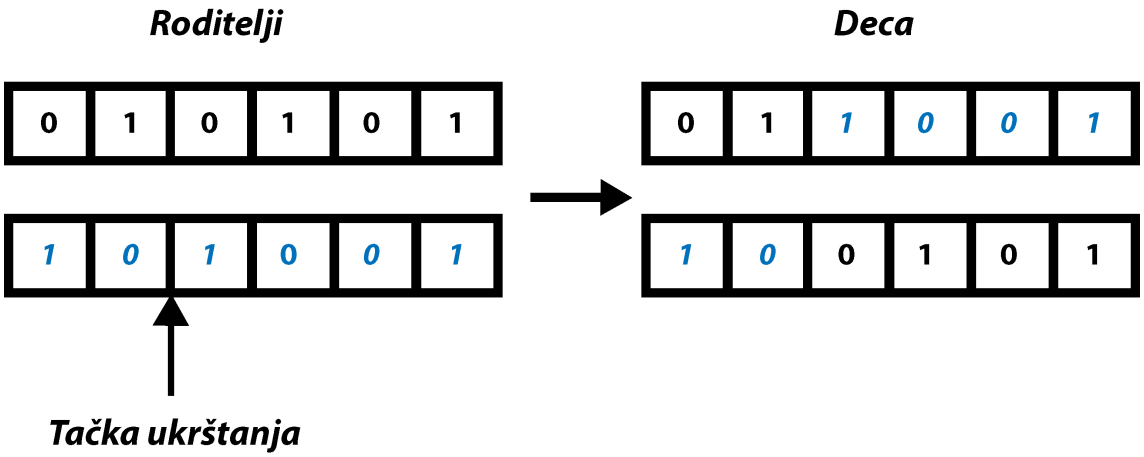
Ukoliko se svako moguće rešenje problema koji je potrebno rešiti može predstaviti nizom bitova, tada bi se za njegovo rešavanje mogao koristiti genetski algoritam. Svako potencijalno rešenje naziva se *kandidatsko rešenje*, ili *jedinka*. Grupa jedinki naziva se *populacija*. Svaki parametar problema se mora kodirati nizom bitova [114].

Neka je npr. jedinka 1 kodirana nizom bitova 010101, a jedinka 2 nizom bitova 101001. Svaki bit se naziva *alelom*. Sekvenca bitova u jedinki koja sadrži informacije o određenoj karakteristici naziva se *gen*. Specifični geni se nazivaju *genotipi*, a

parametar specifičan za problem koji genotip predstavlja naziva se *fenotip*. Skup svih gena u jednoj jedinki naziva se *hromozom*. Jedinka 1 ima hromozom 010101.

Selekcija i ukrštanje

Genetski algoritam može imati veliki broj jedinki, tipično desetine ili stotine. Dve jedinke se mogu spariti, baš kao što to rade jedinke u biološkim populacijama. One se ukrštaju (eng. crossover) što znači da svaka jedinka deli neke svoje genetske informacije sa potomkom. Da bismo pronašli *tačku ukrštanja* (eng. crossover point) za jedinke 1 i 2 koje su kodirane prethodno navedenim nizom bitova, biramo slučajan broj između 1 i 5. Pretpostavimo da izaberemo slučajan broj 2. To znači da dve jedinke razmenjuju sve svoje alele nakon druge pozicije bita u svakom hromozomu (Slika 25) [114].



Slika 25: *Ilustracija ukrštanja kod binarnog GA*

Dva roditelja su se sparila (ukrstila) kako bi stvorila dvoje dece. Svako dete dobija određene genetske informacije od jednog roditelja, a druge genetske informacije od drugog roditelja. Roditelji umiru, a deca preživljavaju da bi nastavila evolutivni proces. Ovaj događaj predstavlja jednu *generaciju* GA. Kao i u biologiji, neka deca će imati visoku prilagođenost, dok će druga imati nisku. Jedinke niske prilagođenosti imaju visoku verovatnoću da ne prežive, tj. uklanjaju se iz simulacije GA. Jedinke sa visokom prilagođenošću preživljavaju da bi se ukrstile sa drugim visoko prilagođenim jedinkama, stvarajući na taj način novu generaciju jedinki. Ovaj proces se nastavlja dok GA ne pronađe prihvatljivo rešenje optimizacionog problema. U nekom trenutku u se mora odlučiti koje jedinke će se ukrštati i razmnožavati. Ova odluka se zasniva na prilagođenosti jedinki u populaciji. Jedinke koje su najbolje prilagođene će se

razmnožavati, dok za jedinke koje su manje prilagođene postoji manja verovatnća da će naći partnere i stoga je verovatno da će umreti bez potomstva. Uobičajeni način za odabir roditelja je ruletski izbor (eng. roulette-wheel selection), koji se naziva i selekcija proporcionalna prilagođenosti (eng. fitness-proportionate selection).

Mutacija

Poslednji korak u GA se naziva *mutacija*. Mutacija je u biologiji relativno retka, tako da je i većini implementacija GA, mutacija takođe retka (reda veličine 2%). Najbolja stopa mutacije zavisi od problema, veličine populacije, kodiranja i drugih faktora. Bez obzira na njenu učestalost, mutacija je važna jer omogućava evolutivnom procesu istraživanje novih potencijalnih rešenja problema. Ako neke genetske informacije nedostaju u populaciji, mutacija pruža mogućnost dodavanja tih informacija u populaciju. Ovo je važno u biološkoj evoluciji, ali je još važnije u GA. GA obično ima populacije male veličine, pa se ukrštanje između srodnika može javiti kao problem.

Da bi se sprovela mutacija, potrebno je odabrati *verovatnoću mutacije*, (npr. 1%). To znači da nakon procesa ukrštanja kojim nastaje potomstvo, svaki bit u svakom nizu koji predstavlja dete ima 1% verovatnoće da pređe u suprotnu vrednost (1 postaje 0, ili 0 postaje 1). Važno je odabrati razumnu stopu mutacije. Previsoka stopa mutacije čini da se GA ponaša kao slučajna pretraga, što nije dobar način za rešavanje problema. Previše niska stopa mutacije dovodi do inbreeding (ukrštanja srodnika) problema, što takođe sprečava GA da pronađe dobro rešenje.

Za populaciju od N jedinki x_i, gde svaka jedinka ima n bitova, i stopu mutacije ρ, na kraju svake generacije svaki bit u svakoj jedinki zameniće se sa verovatnoćom ρ:

$$\begin{aligned} r &\leftarrow U[0,1] \\ x_i(k) &\leftarrow \begin{cases} x_i(k) & ako\ je\ \ r \geq \rho \\ 0 & ako\ je\ \ r < \rho\ \ i\ \ x_i(k) = 1 \\ 1 & ako\ je\ \ r < \rho\ \ i\ \ x_i(k) = 0 \end{cases} \end{aligned} \tag{35}$$

za $i \in [1, N]$ i $k \in [1, n]$, gde je $U[0, 1]$ slučajan broj koji je uniformno raspoređen u intervalu $[0, 1]$ [114].

Algoritam 2 predstavlja pseudo-kod jednostavnog GA.

Algorithm 2 Pseudo-kod jednostavnog GA algoritma

```
Roditelji ← slučajno generisana populacia
while nije zadovoljen kriterijum zaustavljanja do
  Izračunavanje prilagođenosti (podobnosti) svakog roditelja u populaciji
  Potomci← 0
  while |Potomci| < |Roditelji| do
    Na osnovu podobnosti izvršiti izbor para roditelja koji će se razmnožavati
    Ukrštanje roditelja kako bi se dobili potomci c1 i c2
    Potomci← Potomci ⋃ (c1, c2)
    loop
      Slučajna mutacija nekih potomaka
      Roditelji ← Potomci
    end while
end while
Sledeća generacija
```

5.3.3 Simulirano kaljenje

Simulirano kaljenje (eng. Simulated Annealing - SA) je jedna od najjednostavnijih i najpoznatijih metaheurističkih metoda za rešavanje teških globalnih optimizacionih problema [31]. Ranih 1980-ih godina, trojica istraživača Kirkpatrick, Gelatt i Vecchi [72], uvela su koncepte kaljenja u kombinatornu optimizaciju. U početku se ovaj algoritam koristio za rešavanje kombinatornih optimizacionih problema, a vremenom je proširen i na kontinualne optimizacione probleme [119]. Glavna prednost SA je njegova jednostavnost.

Osnovna ideja SA inspirisana je procesom kaljenja koji se koristi u metalurgiji. U ovom procesu, materijal se zagreva, a zatim polako hladi kako bi se postigla snažna kristalna struktura. Kvalitet rezultirajuće strukture zavisi od brzine hlađenja. Ako početna temperatura nije dovoljno visoka ili se primeni brzo hlađenje, dobijaju se nesavršenosti (metastabilna stanja). U tom slučaju, čvrstoća kristala neće dostići termalnu ravnotežu pri svakoj temperaturi. SA algoritam simulira promene energije u sistemu podvrgnutom procesu hlađenja dok ne konvergira ka stanju ravnoteže (stabilnom, zamrznutom stanju) [119]. Ovaj pristup razvio je Metropolis 1953. godine [86].

Ovi koncepti se oslanjaju na snažnu analogiju sa fizičkim kaljenjem materijala. Ovaj proces podrazumeva dovođenje čvrstog materijala do stanja niske energije nakon što mu se podigne temperatura. Može se sažeti u sledeće dva koraka:

• Dovesti čvrsti materijal do veoma visoke temperature dok se ne istopi, odnosno ne pređe u tečno stanje;

• Hlađenje čvrstog materijala prema vrlo specifičnom režimu smanjenja temperature, kako bi se postiglo čvrsto stanje minimalne energije.

U tečnoj fazi, čestice su raspoređene nasumično. Stanje minimalne energije se postiže pod uslovom da je početna temperatura dovoljno visoka i vreme hlađenja dovoljno dugo. Ako to nije slučaj, čvrsti materijal će se naći u metastabilnom stanju sa energijom koja nije minimalna; što se naziva kaljenje i podrazumeva naglo hlađenje čvrstog materijala [72].

Metropolis algoritam

Godine 1953. trojica američkih istraživača (Metropolis, Rosenbluth i Telle [86]) razvila su algoritam za simulaciju fizičkog kaljenja. Njihov cilj bio je verno reprodukovati evoluciju fizičke strukture materijala koji prolazi kroz proces kaljenja. Ovaj algoritam se zasniva na Monte Karlo tehnikama koje se sastoje u generisanju niza stanja čvrstog materijala [72].

Polazeći od početnog stanja i energetskog nivoa E_i, novo stanje j sa energetskim nivoom E_j generiše se modifikacijom položaja jedne čestice. Ako je razlika u energiji, $E_i - E_j$, pozitivna (novo stanje ima nižu energiju), stanje j postaje novo trenutno stanje. Ako je razlika u energiji manja ili jednaka nuli, tada je verovatnoća da stanje j postane trenutno stanje data izrazom:

$$P_r\left\{trenutno\ stanje = j\right\} = e^{-\frac{E_i - E_j}{k_b T}} \tag{36}$$

gde T predstavlja temperaturu čvrstog materijala, a k_b je Boltzmanova konstanta ($k_b = 1,38 \times 10^{-23} J/K$).

Kriterijum prihvatanja novog stanja naziva se Metropolisov kriterijum. Ako se hlađenje vrši dovoljno sporo, čvrsti materijal dostiže stanje ravnoteže pri svakoj zadatoj temperaturi T. U Metropolis algoritmu, ova ravnoteža se postiže generisanjem velikog broja tranzicija pri svakoj temperaturi. Termalna ravnoteža karakteriše se Boltzmannovom statističkom raspodelom. Ova raspodela daje verovatnoću da je

čvrsti materijal u stanju i sa energijom E_i na temperaturi T:

$$P_r\{X = i\} = \frac{1}{Z(T)} e^{-\frac{E_i}{k_b T}} \tag{37}$$

gde je X slučajna promenljiva povezana sa trenutnim stanjem čvrstog materijala, a $Z(T)$ je normalizacioni koeficijent, definisan kao:

$$Z(T) = \sum_{j \in S} e^{-\frac{E_i}{k_b T}} \tag{38}$$

SA algoritam

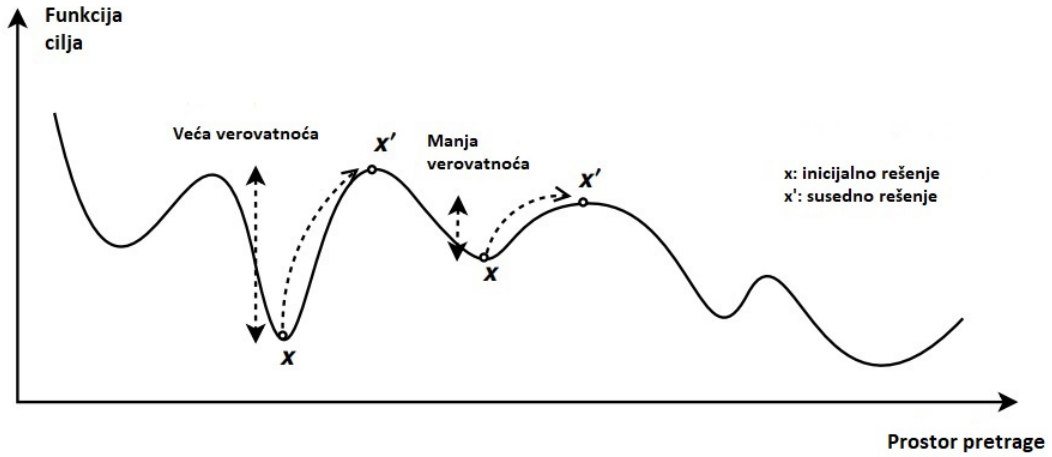
Analogija između fizičkog sistema i optimizacionog problema je sledeća: funkcija cilja problema predstavlja energetsko stanje sistema, rešenje optimizacionog problema analogno je stanju sistema, promenljive odlučivanja povezane sa rešenjem problema odgovaraju pozicijama molekula, dok je globalni optimum analogan osnovnom stanju sistema. Pronalaženje lokalnog minimuma implicira da je postignuto metastabilno stanje. SA spada u stohastičke algoritame i pod određenim uslovima omogućava degradaciju rešenja. Treba izbeći lokalne optimume i time odložiti konvergenciju [119].

SA je algoritam bez memorije, tj. ne koristi nikakve informacije prikupljene tokom pretrage. Iz početnog rešenja, SA napreduje kroz nekoliko iteracija. U svakoj iteraciji se generiše slučajan sused. Potezi koji poboljšavaju funkciju troška uvek se prihvataju. U suprotnom, sused se bira sa određenom verovatnoćom koja zavisi od trenutne temperature i količine degradacije ΔE. ΔE je razlika vrednosti funkcije cilja - energije, trenutnog rešenja i generisanog susednog rešenja. Sa napredovanjem algoritma, verovatnoća prihvatanja takvih poteza se smanjuje. Ova verovatnoća, sledi Boltzmanovu raspodelu:

$$P(\Delta E, T) = e^{-\frac{f(s') - f(s)}{T}} \tag{39}$$

Temperatura se koristi kao kontrolni parametar, da bi se odredila verovatnoća prihvatanja rešenja koja ne poboljšavaju situaciju. Na određenom nivou temperature, istražuje se mnogo pokušaja. Prilikom postizanja ravnotežnog stanja, dolazi do postepenog smanjenja temperature.

Na Slici 26 je prikazan način izbegavanja lokalnog minimuma kod SA. Što je temperatura viša, to je veća verovatnoća da se prihvati loš potez. Pri određenoj temperaturi, što je manje povećanje funkcije cilja, to je veća verovatnoća da se prihvati potez. Bolji potez se uvek prihvata [119].



Slika 26: *Ilustracija izbegavanja lokalnog minimuma kod SA*

Pored trenutnog rešenja, čuva se i najbolje rešenje dobijeno od početka pretrage. Neki parametri kontrolišu napredak pretrage, a to su temperatura i broj iteracija izvršenih pri svakoj temperaturi. Pretraga se može modelovati Markovljevim lancem, gde sledeće stanje zavisi samo od trenutnog stanja. Postoji garancija konvergencije ka optimalnom rešenju:

$$Pr(s_M \in R) \to 1, kada \;\; M \to \infty \tag{40}$$

gde R predstavlja skup globalnih optimalnih rešenja, a s_M rešenje u iteraciji M prema sledećem sporom rasporedu hlađenja:

$$T_k = \frac{\Gamma}{logk} \tag{41}$$

gde je Γ konstanta. U praksi, takav raspored hlađenja je beskoristan jer je konvergencija asimptotska; odnosno, konvergencija se postiže nakon beskonačnog broja iteracija. Međutim, potrebno je mnogo više rada u analizi performansi u konačnom vremenu.

Osim uobičajenih problema u dizajnu S-metaheuristika, kao što su definisanje

okoline i generisanje početnog rešenja, glavni dizajnerski problemi specifični za SA su:

- Funkcija verovatnoće prihvatanja - ključni element SA koji omogućava izbor rešenja koji ne poboljšavaju situaciju.
- Raspored hlađenja - definiše temperaturu u svakom koraku algoritma. Ima ključnu ulogu u efikasnosti algoritma.

$$P_r\left\{prihvatanje\ j\right\} = \begin{cases} 1 & ,ako\ \ je\ \ f(j) < f(i) \\ e^{\frac{f(i)-f(j)}{c}} & ,za\ \ ostale\ \ vrednosti \end{cases} \tag{42}$$

6 Metaheuristike inspirisane inteligencijom rojeva

Algoritmi inspirisani kolektivnim ponašanjem vrsta u prirodi poput mrava, pčela, osa, termita, riba, ptica, itd. nazivaju se algoritmima inteligencije rojeva (eng. Swarm Intelligence Algorithms). Inteligencija rojeva potiče iz socijalnog ponašanja ovih vrsta prilikom njihove potrage za hranom. Glavne karakteristike algoritama zasnovanih na inteligenciji rojeva su:

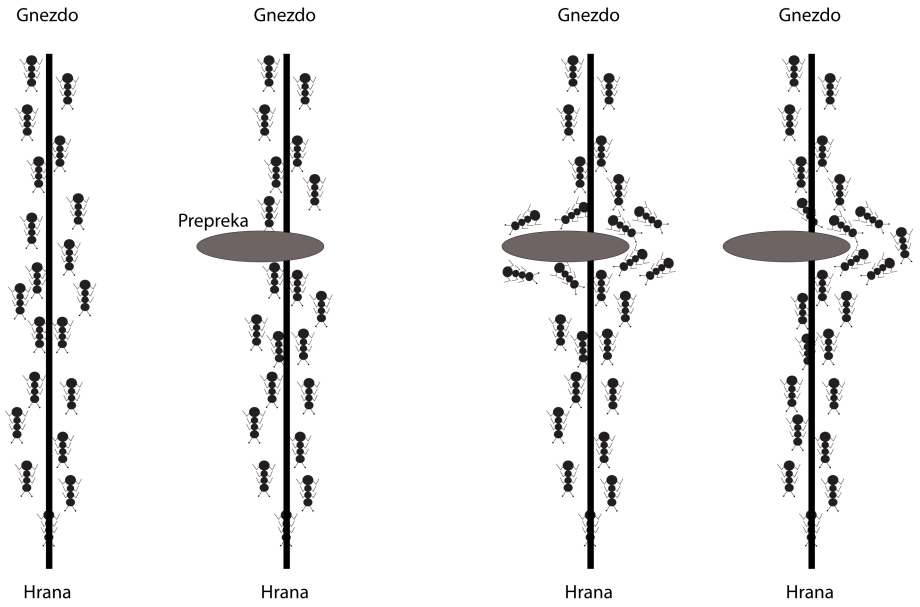
- jedinke su jednostavni i nesofisticirani agenti,
- jedinke međusobno komuniciraju putem indirektne komunikacione veze,
- jedinke se kreću u prostoru pretrage.

Među najuspešnijim optimizacionim algoritmima inspirisanim inteligencijom rojeva su algoritmi kolonije mrava i optimizacije rojem čestica [119].

6.1 Optimizacija mravljom kolonijom

Osnovna ideja u algoritmima optimizacije kolonijom mrava (eng. Ant Colony Optimization - ACO) je da se simulira kooperativno ponašanje pravih mrava, radi rešavanja optimizacih problema. ACO algoritam predložio je Dorigo [34]. Ovaj algoritam se primenjuje na kombinatorne optimizacione probleme i postigao je širok uspeh u rešavanju različitih problema (kao što su raspoređivanje, rutiranje, dodeljivanje, itd)[119].

Pravi mravi obavljaju kompleksne zadatke poput prenošenja hrane i pronalaženja najkraćih puteva do izvora hrane, koristeći kolektivno ponašanje. ACO algoritmi oponašaju princip po kome kolonija mrava pronalazi najkraći put između dve tačke, pri čemu koristi veoma jednostavan mehanizam komunikacije. Na Slici 27 je prikazan eksperiment koji je izvršio Goss [53] sa stvarnom kolonijom argentinskih mrava. Kolonija mrava može stići do izvora hrane, koji je povezan sa kolonijinim gnezdom (mravinjakom), pomoću dva puta. Tokom kretanja kolonije, na zemlji ostaje hemijski trag (feromon). Uloga feromona je da usmeri ostale mrave ka ciljnoj tački. Sa većom količinom ove supstance na putu, povećava se i verovatnoća da će mravi izabrati isti put. Tokom vremena dolazi do smanjenja koncentracije feromona, *proces isparavanja* (eng. evaporation process), a količina koju ostavlja jedan mrav zavisi od količine hrane, *proces pojačavanja* (eng. reinforcement process). Kada se mravi



Slika 27: *Način na koji mravi pronalaze optimalni put od gnezda do hrane*

susretnu sa preprekom (Slika 27), postoji jednaka verovatnoća da svaki mrav izabere levi ili desni put. Pošto je levi put kraći od desnog i zahteva manje vremena, mrav će ostaviti veći nivo feromona na levom putu. Što više mrava odabere levi put, veći je trag feromona. Na taj način se formira najkraći put. Ovaj efekat će biti pojačan fazom isparavanja. Ovakav indirektan način saradnje poznat je kao *stigmergija* (eng. stigmergy).

Algoritam 3 predstavlja osnovni algoritam za ACO.

Algorithm 3 ACO algoritam

Initializacija tragova feromona;
repeat
for svaki mrav **do**
Konstrukcija rešenja korišćenjem tragova feromona;
Ažuriranje tragova feromona;
Proces isparavanja (Evaporation);
Proces pojačavanja (Reinforcement);
until nije zadovoljen kriterijum zaustavljanja
output: Pronađeno je najbolje rešenje, ili skup najboljih rešenja.

Prvo se inicijalizuju informacije o feromonima. Dva iterativna koraka su deo ovog algoritma: konstrukcije rešenja i ažuriranja feromona.

- *Konstrukcija rešenja*: Vrši se prema probabilističkom pravilu prelaska iz stanja u stanje. Veštački mravi mogu se smatrati stohastičkim gramzivim (eng. greedy) procedurama, pri čemu rešenja konstruišu na probabilistički način dodavanjem

komponenti postojećim delimičnim rešenjima, do formiranja potpunog rešenja. Cilj optimizacije se može posmatrati kao graf odlučivanja (ili konstrukcijski graf), u kome mrav konstruiše put. Kod ovog iterativnog procesa uzimaju se u obzir:

- Tragovi feromona: Pamte karakteristike 'dobrih' generisanih rešenja, koja će omogućiti konstruisanje novih rešenja od strane mrava. Tragovi feromona se dinamički menjaju tokom pretrage kao rezultat stečenog znanja. Oni predstavljaju memoriju celokupnog procesa pretrage mrava.

- Informacija o heuristici zavisnoj od problema: Problem-specifične informacije pružaju više smernica mravima u donošenju odluka pri konstrukciji rešenja.

• *Ažuriranje feromona*: Izvodi se pomoću generisanih rešenja. Globalno pravilo ažuriranja feromona primenjuje se u dve faze:

-*Faza isparavanja* je faza u kojoj tragovi feromona automatski opadaju.

Vrednost feromona opada prema jednačini:

$$\tau_{ij} = (1 - \rho)\tau_{ij}, \quad \forall i, j \in [1, n] \tag{43}$$

$\rho \in [0, 1]$ je stopa smanjenja feromona. Isparavanje ima za cilj da se izbegne preuranjeno konvergiranje svih mrava ka dobrim rešenjima i podsticanje diverzifikacije u prostoru pretrage (eksploracija)

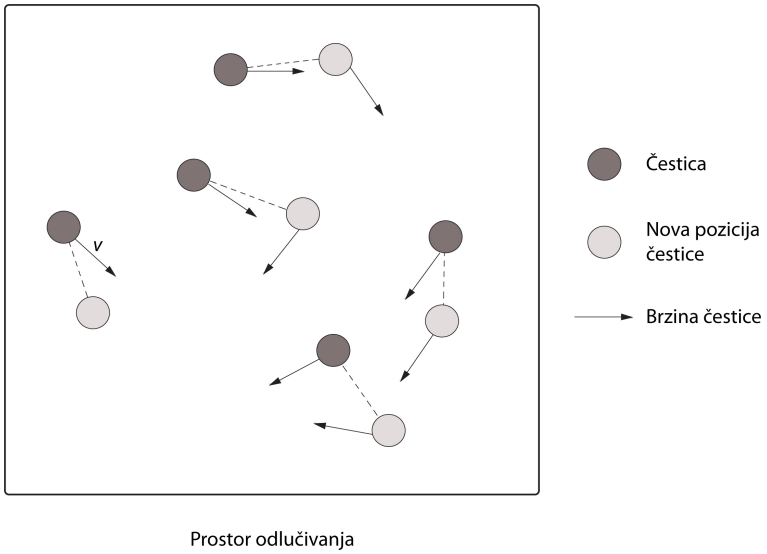
- *Faza pojačavanja* je faza u kojoj se trag feromona ažurira prema generisanim rešenjima.

6.2 Optimizacija rojem čestica

Optimizacija rojem čestica (eng. Particle Swarm Optimization - PSO) je stohastička populaciona metaheuristika inspirisana inteligencijom rojeva. Predstavili su je James Kennedy i Russell C.Eberhart 1995. godine [70]. Ona oponaša socijalno ponašanje prirodnih organizama poput jata ptica i jata riba kako bi pronašli mesto sa dovoljno hrane. U jatima se javlja koordinirano ponašanje koristeći lokalna kretanja, bez centralne kontrole. PSO je prvobitno uspešno osmišljen za kontinualne optimizacione probleme, ali se može adaptirati za diskretne probleme [119].

Roj čini N čestica koje se kreću u D-dimenzionalnom prostoru pretrage. Sve čestice i predstavljaju kandidatska rešenja problema i predstavljene su vektorom x_i

u prostoru pretrage. Svaku česticu karakteriše položaj i brzina, čime je određen smer kretanja i korak čestice (Slika 28).



Slika 28: *Ilustracija načina kretanja čestica u svakoj iteraciji kod PSO [119]*

Optimizacija se postiže kooperacijom između čestica. Uspeh određenih čestica utiče na ponašanje ostalih. Sve čestice sukcesivno prilagođavaju svoj položaj x_i ka globalnom optimumu, uzimajući u obzir sledeće dve komponente:

- Najbolji položaj same čestice (pbesti), označen kao $pi = (p_{i1}, p_{i2}, ..., p_{iD})$.
- Najbolji položaj celog roja (gbest) (ili lbest, najbolji položaj za dati podskup roja), označen kao $p_g = (p_{g1}, p_{g2}, ..., p_{gD})$. Vektor $(p_g - x_i)$ odgovara razlici između trenutnog i najboljeg položaja čestice u njenom okruženju.

Okolina čestica

Za svaku česticu mora se definisati okolina koja označava socijalni uticaj između čestica. Postoji mnogo mogućnosti za definisanje takve okoline. Tradicionalno se koriste dve metode:

- *gbest* metoda - okolina je definisana kao cela populacija čestica
- *lbest* metoda - okolina čestice je skup direktno povezanih čestica. Okolina može biti prazna kada su čestice izolovane (tj. $C2 = 0$).

Lider (*lbest* ili *gbest*) predstavlja česticu koja vodi pretrage čestice ka boljim regionima prostora pretrage.

Svaka čestica je definisana pomoću tri vektora:

- Vektor x sadrži trenutnu poziciju (lokaciju) čestice u prostoru pretrage.
- Vektor p sadrži lokaciju najboljeg pronađenog rešenja do datog trenutka.

- Vektor v sadrži gradijent (smer) kojim će se čestica kretati ako nije ometana.
- Dve vrednosti fitnesa: x-fitnes sadrži fitnes vektora x, dok p-fitnes sadrži fitnes vektora p.

U svakoj iteraciji, primenjuju se sledeće operacije:

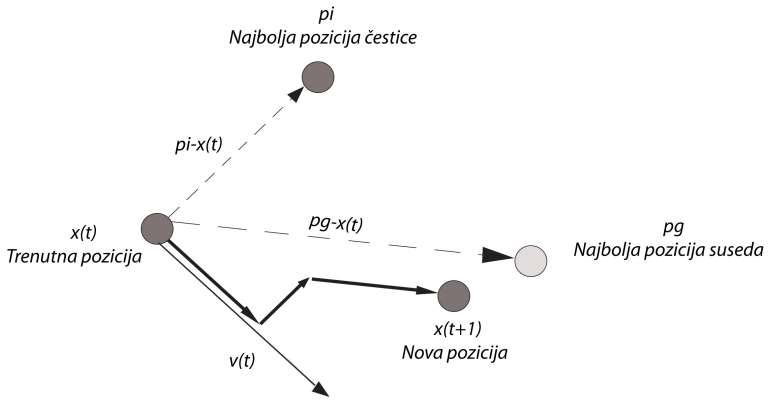
Ažuriranje brzine

Brzina se ažurira prema sledećoj formuli:

$$v_i(t) = v_i(t-1) + \rho_1 C_1 \times (p_i - x_i(t-1)) + \rho_2 C_2 \times (p_g - x_i(t-1)) \tag{44}$$

gde ρ_1 i ρ_2 predstavljaju dve slučajne promenljive u opsegu $[0, 1]$. C_1 i C_2 su konstante koje označavaju faktore učenja. Brzina određuje smer i rastojanje koje čestica treba da pređe, prikazano na Slici 29. Nakon ažuriranja brzine prema formuli, čestica će se kretati oko tačke koja je definisana kao:

$$\frac{\rho_1 p_i + \rho_2 p_g}{\rho_1 + \rho_2} \tag{45}$$



Slika 29: *Način kretanja čestice i ažuriranje brzine*

Elementi vektora v_i su ograničeni na maksimalnu vrednost $[-V_{max}, +V_{max}]$ kako bi se sprečilo preterano naglo kretanje čestica zbog slučajnosti sistema. Ako brzina v_i premaši V_{max} (odn. $-V_{max}$), ona će biti postavljena na V_{max} (odn. $-V_{max}$).

U postupku ažuriranja brzine, prethodnoj brzini se obično dodaje inercijska težina ω:

$$v_i(t) = \omega \times v_i(t-1) + \rho_1 \times (p_i - x_i(t-1)) + \rho_2 \times (p_g - x_i(t-1)) \tag{46}$$

Inercijska težina ω kontroliše uticaj prethodne brzine na trenutnu brzinu. Za velike vrednosti inercijske težine, uticaj prethodnih brzina će biti veći. To znači da

je inercijska težina kompromis između globalne eksploracije i lokalne eksploatacije. Velika inercijska težina podstiče globalnu eksploraciju (tj. raznolikost pretrage u celom prostoru pretrage), dok manja inercijska težina podstiče lokalnu eksploataciju (tj. pojačavanje pretrage u trenutnoj regiji).

Ažuriranje položaja

Koordinate svake čestice u prostoru pretrage se ažuriraju na sledeći način:

$$x_i(t) = x_i(t-1) + v_i(t) \tag{47}$$

Zatim se čestice pomeraju na novu poziciju.

Ažuriranje pozicija najboljih pronađenih čestica

Svaka čestica će (potencijalno) ažurirati najbolje lokalno rešenje:

$$f(x_i) \leq pbest_i \quad \Rightarrow \quad p_i = x_i \tag{48}$$

Osim toga, ažurira se i najbolje globalno rešenje roja:

$$f(x_i) \leq gbest_i \quad \Rightarrow \quad g_i = x_i \tag{49}$$

Algoritam 4 predstavlja osnovni algoritam za PSO.

Algorithm 4 PSO algoritam

```
Sučajna inicijalizacija celog roja čestica;
repeat
  for all čestice i do
    Ažuriranje brzine
    v_i(t) = v_i(t-1) + ρ1C1 × (p_i − x_i(t-1)) + ρ2C2 × (p_g − x_i(t-1))
    Pomeranje čestice na novu poziciju: x_i(t) = x_i(t-1) + v_i(t)
    if f(x_i) ≤ f(pbest_i) then
      pbest_i = x_i
      if f(x_i) ≤ f(gbest_i) then
        gbest_i = x_i
        Update (x_i,v_i)
      end if
    end if
  end for
until Nije zadovoljen kriterijum zaustavljanja
```

6.3 Optimizacija veštačkom kolonijom pčela

U prirodi postoji više od 20 000 poznatih vrsta pčela. One naseljavaju cvetna područja i žive u socijalnim kolonijama u svojim gnezdima koja se nazivaju košnice. U jednoj košnici živi između 60 000 i 80 000 jedinki. Pčele proizvode kompleksnu supstancu koja se naziva med, a svoje košnice grade uz pomoć voska. Kao izvor energije, pčele koriste nektar, a za odgoj potomaka kao izvor proteina, koriste polen.

Uopšteno, svaka kolonija pčela sadrži jednu reproduktivnu ženku, koja se naziva matica. Pored matice u koloniji postoji nekoliko hiljada mužjaka - trutova i više hiljada sterilnih ženki – radilica. Nakon parenja sa nekoliko trutova, matica izleže veliki broj mladih pčela koje čine potomstvo.

Ponašanje pčela prilikom sakupljanja hrane, izbora mesta za gnezdo i njihovo ponašanje prilkom parenja, glavne su aktivnosti u životu kolonije pčela koje se koriste prilikom dizajniranja algoritama za optimizaciju [119].

Prvi algoritam za optimizaciju koji je inspirisan veštačkom kolonijom pčela (eng. Artificial Bee Colony - ABC), predstavio je Karaboga 2005. godine [69].

Pčele koje pripadaju koloniji se u proleće, ili rano leto, podele u dve grupe. Jednu grupu čini matica i polovina pčela radilica, a drugu ćerka matica i druga polovina radilica. Prvi skup ima za cilj da osnuje novu koloniju, dok drugi nastavlja svoj život sa starom kolonijom. Da bi pronašle novo mesto za gnezdo, nekoliko stotina pčela ima funkciju izviđača, dok ostale pčele miruju kako bi sačuvale energiju roja. Kada se izabere novo mesto, tada se sve pčele roja sele. Pčele koje sakupljaju hranu, svojim plesom pokazuju različita mesta za novo gnezdo. Postoji osam različitih plesova, koji se nazivaju plesovi treperenja (eng. waggle dance), a brzina i orijentacija plesa povezane su sa kvalitetom mesta za gnezdo. Za karakterizaciju mesta koriste se različiti atributi kao što su: površina ulaza, smer ulaza, visina ulaza, zapremina šupljine itd. Vremenom se ples pčela fokusira na jedno mesto, tako što pčele izviđači 'glasaju' plesom za svoje omiljeno mesto, a zatim se donosi grupna odluka i čitav skup pčela se seli na novo mesto.

Ples treperenja se koristi kako bi se druge pčele obavestile o lokaciji mesta za gnezdo, ili izvorima hrane. Potraga za mestom za gnezdo, slična je donekle sa traženjem izvora hrane. Izvor hrane se otkriva putem plesa treperenja, a zatim se vrši iskorišćavanje nektara. Kvalitet izvora hrane zavisi od bogatstva energijom,

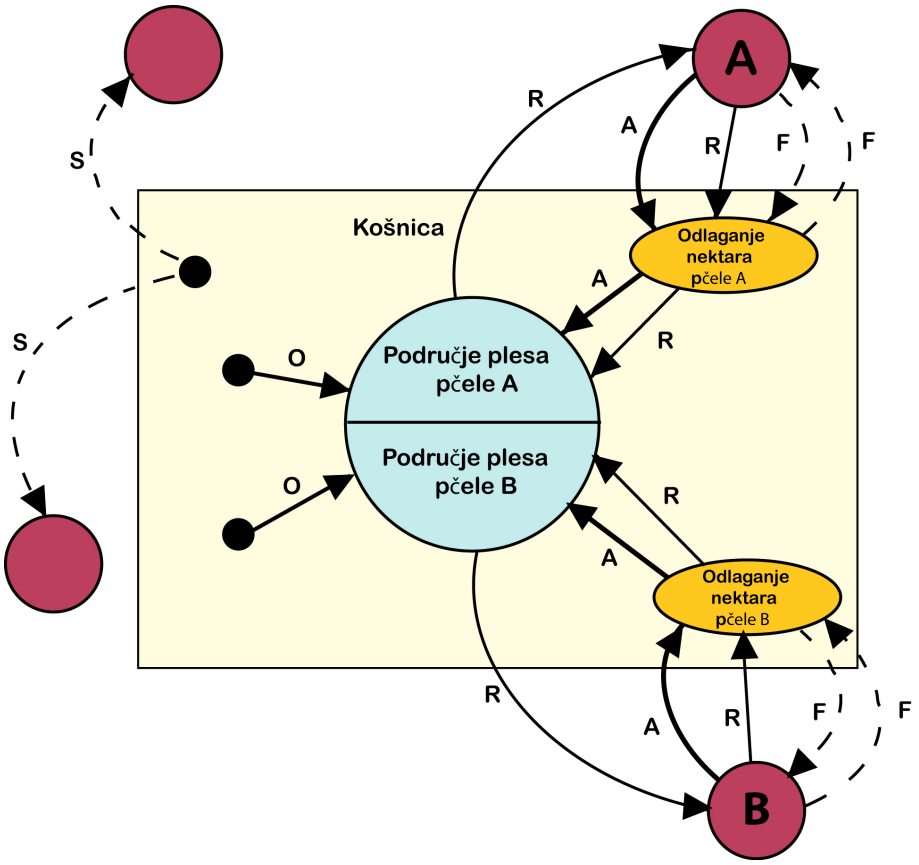
udaljenosti od gnezda i težine ekstrakcije energije.

Za formiranje kolektivnog znanja, najvažnija je razmena informacija između pčela. Najvažniji deo svake košnice za razmenu informacija je područje plesa. Tu se odvija komunikacija između pčela koja je vezana za kvalitet izvora hrane. Ona se odvija uz pomoć plesa treperenja. Posmatrač može da se, na osnovu svih plesova koje posmatra na podijumu, odluči za zapošljavanje na najprofitabilnijem izvoru. Više informacija se širi o izvorima koji su profitabilniji, pa je veća verovatnoća da će se posmatrači odlučiti baš za njih. Zaposleni sakupljači uz pomoć plesa dele svoje informacije sa verovatnoćom koja je proporcionalna profitabilnosti izvora hrane, a deljenje ovih informacija pomoću plesa duže traje. Iz tog razloga je regrutovanje proporcionalno profitabilnosti izvora hrane [67].

Na Slici 30 je prikazano ponašanje kolonije pčela nakon otkrivanja izvora hrane. Pretpostavljamo da su pronađena dva izvora hrane, A i B. Nezaposlena pčela nema informacije o izvoru hrane. To može biti pčela izviđač S (eng. scout) koja počinje da istražuje okolinu gnezda ili pčela posmatrač O (eng. onlooker), koja prati plesove pčela. Nakon lokalizacije izvora hrane, pčela postaje zaposlena pčela. Nakon što odloži nektar, zaposlena pčela ima tri mogućnosti: napuštanje izvora hrane (A), regrutovanje drugih pčela iz gnezda (R) i sakupljanje hrane bez regrutovanja drugih pčela (F).

Prilikom procesa sakupljanja hrane kolonija pčela ima za cilj da pronađe izvore bogate hranom, kako bi dobila maksimalna količina nektara. Pčele sakupljači (radilice) se, prema ulozi koje imaju u ovom procesu, mogu podeliti u na nekoliko vrsta:

- Nezaposleni sakupljači - Pčele koje nemaju nemaju informaciju o izvorima hrane. One traže izvor hrane za ekspolataciju. Postoje dve vrste nezaposlenih pčela : izviđači (pčele koje istražuju okolinu gnezda u potrazi za izvorom hrane) i posmatrači (čekaju u gnezdu i prate plesove treperenja drugih pčela, kako bi otkrile izvore hrane).
- Zaposleni sakupljači – Ove pčele imaju dovoljno informacija o izvoru hrane. One ga pronalaze i iskorišćavaju, pamte njegovu lokaciju, nose deo nektara za košnicu i tamo ga skladište. Količina nektara može imati tri stanja. Ukoliko je količina niska, ili iscrpljena, ove pčele napuštaju izvor hrane i postaju nezaposleni pčele. Ukoliko je količina nektara dovoljna, ove pčele nastavljaju da ga sakupljaju, ali ne dele izvor hrane sa drugim pčelama iz gnezda. Treći slučaj je da su pčele pronašle značajnu



Slika 30: *Ponašanje kolonije pčela nakon otkrivanja izvora hrane [119]*

količinu nektara i tada one izvode ples treperenja kako bi privukle druge pčele iz gnezda.

Potraga za izvorom hrane zasnovana je na dve strategije:

- Eksploracija (istraživanje) izvora hrane: Pčela izviđač istražuje okolinu kako bi pronašla izvor hrane. Ukoliko ga je pronašla, ona se vraća na "plesni podijum" košnice i tamo plesom obaveštava ostale pčele o tome. Pčele posmatrači tada postaju pčele za regrutaciju, a zatim postaju zaposlene pčele.
- Eksploatacija (iskorišćavanje) izvora hrane: Zaposlena pčela sakupljač izračunava količinu hrane u izvoru i na osnovu kvaliteta nektara donosi odluku o tome da li će nastaviti njegovu eksploataciju i zapamtiti najbolju lokaciju do tada, ili će ga napustiti. Ako napusti izvor, ova pčela postaje nezaposleni izviđač, ili posmatrač.

U Tabeli 1 data je analogija između prirodne i veštačke kolonije pčela.

Kolinija pčela se može prostirati na velike udaljenosti u više pravaca, kako bi ekspolatisala što veći broj izvora hrane. Kolonija raspoređuje svoje pčele tako što cvetne površine sa većom količinom nektara i polena posećuje veći broj pčela, dok

Tabela 1: *Analogija između prirodne i veštačke kolonije pčela*

Prirodna kolonija pčela	Veštačka kolonija pčela
Izvor hrane	Rešenje
Kvalitet nektara	Funkcija cilja
Posmatrači	Eksploatacija pretrage
Izviđači	Eksploracija pretrage

manje bogate površine privlače manji broj pčela. Proces sakupljanja započinje tako što pčele izviđači istražuju cvetne površine koje obećavaju. Pčele izviđači se slučajno kreću sa jedne površine na drugu. Tokom sakupljanja, kolonija nastavlja sa daljim istraživanjem, a određeni procenat populacije i dalje čine pčele izviđači. Kada se vrate u košnicu, pčele koje su pronašle površinu sa kvalitetom nektara, koji je iznad nekog određenog praga, ostavljaju svoj nektar i polen i izvode ples treperenja. Ovaj ples je ključan za komunikaciju između pčela i sadrži tri vrste informacija o cvetnoj površini: smer u kome se nalazi, njena udaljenost od košnice i kvalitet. Na ovaj način kolonija šalje svoje pčele tačno ka toj površini. Znanje svake pčele o okolini, dobija se isključivo iz plesa treperenja. On omogućava koloniji da proceni kvalitet cvetnih površina koji zavisi od kvaliteta hrane i količine energije potrebne za njenu berbu. Nakon plesa, pčela izviđač se vraća na cvetnu površinu zajedno sa pčelama koje su čekale unutar košnice. Na površine koje više obećavaju, šalje se veći broj pčela, što omogućava koloniji da brzo i efikasno prikupi hranu. Dok sakupljaju hranu, pčele vode računa o nivou preostale hrane, jer od toga zavisi njihov ples pri povratku u košnicu. Ukoliko je izvor hrane i dalje bogat hranom, nakon plesa treperenja, još veći broj pčela će biti regrutovan. Regrutacija je proporcionalna kvalitetu izvora hrane.

Algoritam 5 predstavlja osnovni ABC algoritam.

Algorithm 5 ABC algoritam

```
Sučajna inicijalizacija kolonije pčela;
repeat
    Radilice se postavljaju na izvore hrane i određuje se količina nektara;
    Posmatrači se postavljaju na izvore hrane i određuje se količina nektara;
    Izviđači idu u potragu za novim izvorima hrane.
    Pamti se najbolji izvor hrane koji je pronađen;
until Nije zadovoljen kriterijum zaustavljanja
```

Pčele traže izvore hrane na takav način kako bi se dobio maksimalan odnos E/T (gde je E dobijena energija, a T vreme provedeno u sakupljanju hrane). U slučaju

rojeva pčela, E je proporcionalno količini nektara u izvoru hrane koji pčele otkriju. Roj pčela radi na maksimizaciji količine meda koja se skladišti unutar košnice. U zadatku maksimizacije, cilj je pronaći maksimum funkcije cilja $F(\theta)$, gde je $\theta \in R^p$. Pretpostavimo da je θ_i pozicija i-tog izvora hrane; $F(\theta_i)$ predstavlja količinu nektara izvora hrane koji se nalazi na poziciji θ_i i proporcionalna je energiji $E(\theta_i)$. Neka $P(c) = \{\theta_i(c)|i = 1, 2, ..., S\}$ (c- ciklus, S- broj izvora hrane oko košnice) predstavlja populaciju izvora hrane koje su pčele posetile [67].

Za koji izvor hrane će se odlučiti pčela posmatrač, zavisi od količine nektara $F(\theta)$ tog izvora hrane. Sa porastom količine nektara, verovatnoća da će pčela odabrati taj izvor hrane povećava se proporcionalno. Stoga se verovatnoća da će pčela odabrati izvor hrane koji se nalazi na poziciji θ_i može izraziti kao:

$$P_i = \frac{F(\theta_i)}{\sum_{k=1}^{S}(F(\theta_i))} \tag{50}$$

Nakon posmatranja plesova zaposlenih pčela, pčela posmatrač odlazi do regije izvora hrane koji se nalazi na poziciji θ_i i bira jedan od izvora hrane nakon što napravi poređenje između izvora hrane oko ove pozicije. Pozicija izabranog susednog izvora hrane se izračunava na sledeći način:

$$\theta_i(c+1) = \theta_i(c) \pm \phi_i(c) \tag{51}$$

$\phi_i(c)$ predstavlja nasumično generisan korak kako bi se pronašao izvor hrane s više nektara oko θ_i. $\phi(c)$ se izračunava kao razlika između $\phi_i(c)$ i $\phi_k(c)$ (gde je k nasumično generisan indeks) pozicija hrane. Ukoliko je količina nektara $F(\theta_i(c+1))$ na poziciji $\theta_i(c+1)$ veća od one na $\theta_i(c)$, tada pčela odlazi do košnice i deli svoje informacije s drugima, pozicija $\theta_i(c)$ izvora hrane se menja u $\theta_i(c+1)$, dok $\theta_i(c)$ ostaje nepromenjen.

Svaki izvor hrane ima samo jednu zaposlenu pčelu. Zbog toga je broj zaposlenih pčela jednak broju izvora hrane. Ako se pozicija θ_i izvora hrane i ne može poboljšati u unapred određenom broju pokušaja koji predstavljaju 'limit', tada taj izvor hrane θ_i napušta zaposlena pčela, koja postaje izviđač. Izviđač počinje sa traženjem novog izvora hrane, i nakon što pronađe novi izvor, nova pozicija se prihvata kao θ_i .

6.4 Optimizacija inspirisana ponašanjem svitaca

U prirodi postoji oko dve hiljade vrsta svitaca. Većina svitaca proizvodi kratke, ritmične treptaje, a obrazac treptanja često je jedinstven za određenu vrstu. Prava funkcija svetlosti koju svici proizvode, još uvek nije jasno ustanovljena. ali se smatra da su razlozi njenog postojanja: privlačenje partnera za parenje (komunikacija) i privlačenje potencijalnog plena. Takođe, ova svetlost može služiti i kao mehanizam zaštitnog upozorenja potencijalnom predatoru, na gorak ukus svitaca.

Intenzitet svetlosti I, na određenoj udaljenosti r od svetlosnog izvora, opada po zakonu $I \sim 1/r^2$, sa povećanjem udaljenosti r. Vazduh apsorbuje svetlo, pa ono postaje sve slabije sa porastom rastojanja r. Kombinacija ova dva faktora čini da je većina svitaca vidljiva samo noću na udaljenosti od nekoliko stotina metara, što je obično dovoljno za njihovu međusobnu komunikaciju.

Treptava svetlost je povezana sa funkcijom cilja koja treba da se optimizuje, što omogućava formulisanje novih algoritama za optimizaciju.

Optimizacioni algoritam inspirisan ponašanjem svitaca (engl. Firefly algorithm - FA), definisao je Xin-She Yang na Univerzitetu u Cambridge-u 2007. godine [131].

Prilikom razvoja algoritma inspirisanog svicima neke karakteristike treptanja svitaca morale su biti idealizovane. Pritom su korišćena sledeće tri pravila:

1. Svi svici su jednog pola, tako da će jedan svitac biti privučen drugim svicima bez obzira na njihov pol.

2. Privlačnost je proporcionalna intenzitetu svetlosti, tako da će se uvek svitac koji emituje svetlost nižeg intenziteta kretati prema svicu koji emituje svetlost većeg intenziteta. Privlačnost je proporcionalna intenzitetu emitovane svetlosti i obe opadaju sa povećanjem udaljenosti. Ako ne postoji svitac koji emituje svetlost većeg intenziteta od posmatranog svica, on će se kretati nasumično.

3. Jačina svetlosti koju emituje svitac zavisi, ili je potpuno odrđena raspodelom funkcije cilja.

Pseudo-kod FA dat je Algoritmom 6

Algorithm 6 Pseudo-kod FA

```
Definisanje funkcije cilja f(x), x = (x_1, ..., x_d)^T
Generisanje inicijalne populacije svitaca x_i (i = 1, 2, ..., n)
Intenzitet svetlosti I_i u x_i određen je funkcijom f(x_i)
Definisanje koeficijenta apsorpcije svetlosti γ
while (t < MaxGeneration) do
  for i = 1 : n  za  svih  n  svitaca do
    for j = 1 : n  za  svih  n  svitaca do
      if (I_i < I_j) then
        Svitac i se pomera prema svicu j
      end if
      Promena privlačnosti sa rastojanjem r prema zakonu exp[−γr]
      Prcena novih rešenja i ažuriranje intenziteta svetlosti
    end for  j
  end for  i
  Rangiranje jedinki i pronalaženje trenuto najboljeg rešenja g*
end while
Obrada rezultata i vizualizacija
```

Promena intenziteta svetlosti koju emituje svitac i formulacija privlačnosti su dva važna pitanja u FA. Za probleme maksimalne optimizacije, jačina svetlosti I koju emituje svitac na određenoj poziciji x može biti određena kao $I(x) \approx f(x)$. S obzirom da je privlačnost β relativna veličina, ona će se menjati sa udaljenošću r_{ij} između svetaca i i j. Pošto intenzitet emitovane svetlosti opada sa udaljavanjem od izvora, a svetlost biva apsorbovana u prostoru kroz koji se kreće, privlačnost svica će se menjati u zavisnosti od stepena apsorpcije.

U najjednostavnijem obliku, intenzitet svetlosti $I(r)$, opada po inverznom kvadratom zakonu

$$I(r) = \frac{I_s}{r^2} \tag{52}$$

gde je I_s intenzitet na samom izvoru svetlosti. Intenzitet svetlosti I se menja sa rastojanjem r od izvora, po sledećem zakonu:

$$I = I_0 e^{-\gamma r} \tag{53}$$

gde je I_0 originalni intenzitet svetlosti, dok je γ koeficijent apsorpcije sredine kroz koju se svetlost prostire. Kako bi se u izrazu (52) izbegla vrednost $r = 0$, koristi se Gausov oblik:

$$I(r) = I_0 e^{-\gamma r^2} \tag{54}$$

Kako je privlačnost svica proporcionalna intenzitetu svetlosti koju vide susedni svici, ona se može definisati izrazom:

$$\beta = \beta_0 e^{-\gamma r^2} \tag{55}$$

gde je β_0 privlačnost na udaljenosti $r = 0$. Zbog lakšeg izračunavanja i izbegavanja eksponencijalne funkcije, prethodni izraz se može aproksimirati na sledeći način:

$$\beta = \frac{\beta_0}{1 + Cr^2} \tag{56}$$

Izrazi (55) i (56) definišu karakterističnu udaljenost $\Gamma = 1/\sqrt{\gamma}$. Funkcija privlačnosti $\beta(r)$ može biti bilo koja monotono opadajuća funkcija, kao što je dato u sledećem izrazu:

$$\beta(r) = \beta_0 e^{-\gamma r^m}, \quad (m \geq 1) \tag{57}$$

Za fiksno γ, Γ postaje

$$\Gamma = \gamma^{-\frac{1}{m}} \to 1, \quad m \to \infty \tag{58}$$

Suprotno, za datu dužinu Γ u nekom optimizacionom problemu, parametar γ se može koristiti kao tipična inicijalna vrednost:

$$\gamma = \frac{1}{\Gamma^m} \tag{59}$$

Međusobno rastojanje između bilo koja dva svica i i j koji se nalaze na pozicijama x_i i x_j, respektivno, može se izračunati pomoću Euklidske udaljenosti:

$$r_{ij} = ||x_i - x_j|| = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \tag{60}$$

gde je $x_{i,k}$ k-ta komponenta prostorne koordinate i-tog svica. U 2-D slučaju, imamo

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{61}$$

Kretanje svica i, koji je privučen drugim privlačnijim (svetlijim) svicem j, može se

predstaviti sledećim izrazom:

$$x_i = x_i + \beta_0 e^{-\gamma r_{ij}^2}(x_j - x_i) + \alpha\epsilon_i \tag{62}$$

gde drugi sabirak definiše promenu privlačnosti. α je parametar slučajnosti, ϵ_i je vektor slučajnih brojeva dobijenih Gausovom, ili uniformnom raspodelom. Najjednostavnije je da se ϵ_i zameni sa $rand - 1/2$, gde je $rand$ slučajni broj generisan uniformnom raspodelom u opsegu [0,1]. Najjčešće je $\beta_0 = 1$, dok je $\alpha \in [0,1]$.

Parametar γ u teoriji može da ima vrednost iz opsega $[0, \infty)$, ali je u praksi određen karakterističnom dužinom Γ sistema koji se optimizuje, kao $\gamma = O(1)$, a njegove tipične vrednosti se menjaju u opsegu od 0.1 do 10.

7 Prikaz implementacije hibridnih metoda metaheuristika i modela rekurentnih neuronskih mreža

U ovom poglavlju teze prikazan je originalni algoritam pretrage reptila (eng. Reptile Search Algorithm - RSA) i predložena je njegova modifikovana verzija unapređeni algoritam pretrage reptila (eng. Enhanced Reptile Search Algorithm - ERSA). Opisane su tehnike dekompozicije signala VMD i EMD, metrike koje se koriste za ocenu performanisi modela, kao i SHAP analiza. Prikazani su i rezultati dobijeni testiranjem predloženog ERSA na standardnim benčmark funkcijama (CEC2019). Dat je prikaz dva korišćena skupa podataka o proizvodnji vetra, na kojima su primenjene VMD i EMD tehnike dekompozicije. Na kraju je prikazan i detaljni tok eksperimentalnog procesa.

7.1 Originalni algoritam pretrage reptila (RSA)

RSA je optimizacioni algoritam koji matematički simulira predatorske tehnike krokodila. Pri lovu krokodila postoje dve glavne tehnike: opkoljavanje i lov. Detalji obe tehnike su opisani u originalnom algoritmu [2].

7.1.1 Ponašanje krokodila u prirodi

Krokodili su veliki poluvodeni gmizavci koji žive širom tropskih područja Australije, Afrike, Azije i Amerike. Njihove fizičke karakteristike omogućavaju im da budu snažni predatori, a njihovo telo u obliku strele, povećava njihovu brzinu i olakšava kretanje kroz vodu i vazduh. Prilikom hodanja oni podižu svoje noge sa strane, što im omogućava brže kretanje.

Krokodili imaju savršen noćni vid i uglavnom su noćni lovci. Oni love ribe, gmizavce, rakove, vodozemce, mekušce, sisare i ptice, a ponekad jedu i manje krokodile. Njihov usporeni metabolizam im omogućava da prežive dugo vremena bez hrane. Iako se čini da se kreću sporo, krokodili su brzi lovci i moćni predatori u svom staništu, a vrlo često napadaju i ubijaju druge grabljivice.

Krokodili mogu trčati veoma brzo na kratkim udaljenostima, čak i van vode.

Kada se brzo kreću, oni drže noge u ravnom i uspravnom položaju ispod tela. Ovaj način hodanja se naziva visoki hod i omogućava krokodilima veliku brzinu. Krokodili poseduju neke napredne kognitivne sposobnosti, tako da mogu prepoznati i iskoristiti obrasce ponašanja plena.

Krokodili su napredni lovci i često sarađuju kao tim kada love plen. Lov se obavlja kooperativno među krokodilima na osnovu koordinacije i saradnje. Koordinirani lov je složen oblik saradničkog lova, u kome određeni predatori prate kretanje ciljnog plena tokom organizovanog lova. Na ovaj način se određeni predatori udružuju u međusobnim akcijama.

Koordinacija i saradnja među krokodilima tokom lova je fascinantna. Oni deluju kao tim, a mogu formirati veoma inteligentne i koordinirane lovne grupe u kojima pojedinci preuzimaju različite uloge. Njihova strategija lova je složena, često uključujući zasede i iskorišćavanje sporog metabolizma, pri čemu se većina lova odvija noću u plitkim vodama. Na primer, krokodili mogu zajedno delovati kako bi prisilili ribe da se okupe u grupu. Zatim se smenjuju u lovu na ribe. Krokodili različitih veličina mogu preuzeti različite uloge, pri čemu veći krokodili teraju ribe iz dublje vode prema plićim područjima, dok manji krokodili, koji su agilniji, sprečavaju njihovo bekstvo.

Postoje slučajevi kada krokodil može uplašiti plen prisiljavajući ga da brzo uđe u lagunu gde drugi krokodili čekaju da ga napadnu. Nakon uspešnog lova, krokodil će se udaljiti iz lovnog područja i ponovo se pridružiti grupi nakon što pojede svoj plen.

Može se reći da su krokodili jedni od najinteligentnijih i najveštijih lovaca posle ljudi. Ponašanje krokodila je matematički modelirano kako bi se odredilo najbolje rešenje uzimajući u obzir specifična ograničenja.

7.1.2 Faza inicijalizacije

Optimizacija započinje slučajno generisanim kandidatima (X), kao što je prikazano sledećim jednačinama:

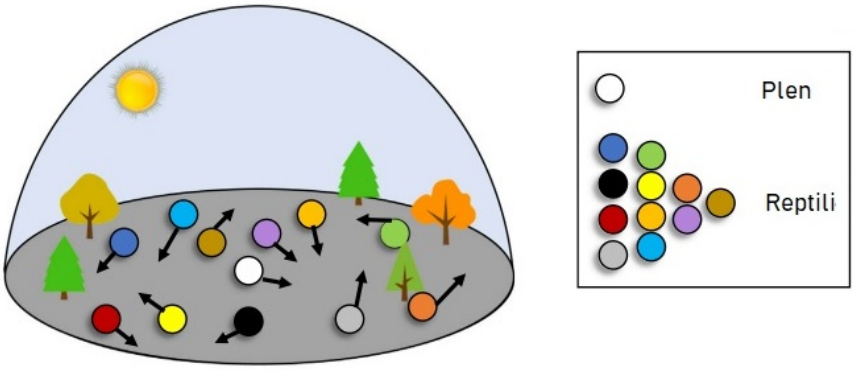
$$X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & x_{1,n-1} & x_{1,n} \\ x_{2,1} & \cdots & x_{2,j} & \cdots & x_{2,n} \\ \cdots & \cdots & x_{i,j} & \cdots & \cdots \\ \vdots & & \vdots & \vdots & \vdots \\ x_{N-1,1} & \cdots & x_{N-1,j} & \cdots & x_{N-1,n} \\ x_{N,1} & \cdots & x_{N,j} & x_{N,n-1} & x_{N,n} \end{bmatrix} \tag{63}$$

$$x_{i,j} = rand \times (UB - LB) + LB, j = 1, 2, ..., n \tag{64}$$

gde $x_{i,j}$ predstavlja j-tu lokaciju i-tog agenta, N je broj potencijalnih agenata i n označava dimenziju za dati problem. U jednačini (64), $rand$ je slučajna vrednost iz uniformne raspodele, LB predstavlja donju granicu, a UB gornju granicu za dati problem. Tokom nezavisnih testiranja razmatrane su različite raspodele. Najbolje rezultate dala je upotreba uniformne raspodele.

7.1.3 Faza opkoljavanja (exploracija)

Tokom faze opkoljavanja (Slika 31), krokodili se kreću na dva načina: uspravan hod (eng. high walking) i hodanje po stomaku (eng. belly walking). Kod RSA postoje dve faze pretrage: *opkoljavanje* (eksploracija) i *lov* (eksploatacija). Promena se vrši na osnovu četiri kriterijuma i određuje se na osnovu trenutne iteracije.



Slika 31: *Opkoljavanje plena, kada je* $(t \leq \frac{T}{2})[3]$

Ažuriranje pozicije za fazu eksploracije prikazano je u jednačini (65).

$$x_{(i,j)}(t+1) = \begin{cases} Best_j(t) \times -\eta_{(i,j)}(t) \times \beta - R_{(i,j)}(t) \times rand, & t \leq \frac{T}{4} \\ Best_j(t) \times x_{(r_1,j)} \times ES(t) \times rand, & t \leq 2\frac{T}{4} \ and \ t > \frac{T}{4} \end{cases} \tag{65}$$

gde $Best_j(t)$ predstavlja j-tu lokaciju trenutnog optimalnog agenta, $rand$ označava proizvoljan broj u opsegu $[0, 1]$, t predstavlja trenutni broj iteracije, dok je T maksimalan broj iteracija. Vrednost $\eta_{(i,j)}$ se izračunava prema jednačini (66), i opisuje operator lova za j-tu poziciju u i-tom rešenju. Vrednost β se koristi kako bi se kontrolisala osetljivost. $R_{(i,j)}$ je funkcija redukcije (vrednost koja se koristi za smanjenje oblasti pretrage), prikazana u jednačini (67). Vrednost r_1 se proizvoljno bira iz opsega $[1, N]$, promenljiva $x_{(r_1,j)}$ označava proizvoljnu lokaciju u i-tom rešenju. $ES(t)$ (eng. Evolutionary Sense) je verovatnoća sa slučajno opadajućim vrednostima u opsegu $[2, -2]$, koja se računa prema jednačini (68).

$$\eta_{(i,j)} = Best_j(t) \times P_{(i,j)} \tag{66}$$

$$R_{(i,j)} = \frac{Best_j(t) - x_{(r_2,j)}}{Best_j(t) + \epsilon} \tag{67}$$

$$ES(t) = 2 \times r_3 \times \left(1 - \frac{1}{T}\right) \tag{68}$$

gde $P_{(i,j)}$ definiše razlike u procentima između j-tog i najbolje dostignutog j-tog rešenja, što se određuje putem jednačine (69). Promenljiva ϵ obezbeđuje minimalnu vrednost koja se koristi kako bi se izbeglo deljenje nulom. r_2 i r_3 su proizvoljne vrednosti između $[1, N]$ i $[-1, 1]$, respektivno.

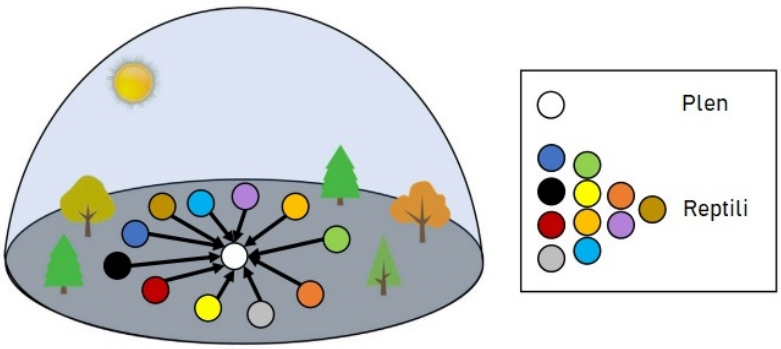
$$P_{(i,j)} = \alpha + \frac{x_{(i,j)} - M(x_i)}{Best_j(t) \times (UB_{(j)} - LB_{(j)}) + \epsilon} \tag{69}$$

gde je $M(x_i)$ predstavlja srednju lokaciju i-tog agenta, određenu prema jednačini (70). $LB(j)$ i $UB(j)$ su donje i gornje granice za j-tu lokaciju, dok se promenljiva α koristi za regulisanje osetljivosti.

$$M(x_i) = \frac{1}{n}\sum_{j=1}^{n} x_{(i,j)} \tag{70}$$

7.1.4 Faza lova (eksploatacija)

Mehanizmi eksploatacije RSA, prikazani su tokom faze lova (Slika 32).



Slika 32: *Napad na plen, kada je* $(t > \frac{T}{2})$ *[3]*

Tokom procesa lova, krokodili izvode, koordinaciju lova ili saradnju (kooperaciju) u lovu. U skladu sa tim, ažuriranje pozicija za fazu eksploatacije je prikazano jednačinom (71).

$$x_{(i,j)}(t+1) = \begin{cases} Best_j(t) \times -P_{(i,j)}(t) \times rand, & t \leq 3\frac{T}{4} \ \ and \ \ t > 2\frac{T}{4} \\ Best_j(t) - \eta_{(i,j)}(t) \times \epsilon - R_{(i,j)}(t) \times rand, & t \leq T \ \ and \ \ t > 3\frac{T}{4} \end{cases} \tag{71}$$

gde $Best_j(t)$ predstavlja j-tu lokaciju do sada najboljeg dobijenog agenta, ϵ je mala vrednost, $P_{(i,j)}$, $\eta_{(i,j)}$ i $R_{(i,j)}$ su dati jednačine (69), (66) i (67), respektivno.

Pseudo-kod RSA dat je Algoritmom 7.

Algorithm 7 Pseudo-kod RSA

```
Setovanje vrednosti parametara
Formiranje populacije X
while (t < T) do
  Izračunati fitnes funkciju svih agenata i sačuvati najbolju
  Preračunati ES
  for Agenta u populaciji do
    for Agenta u sub-populaciji do
      Izračunati η, R, P
      if (t ≤ T/4) then
        Primeniti strategiju uspravnog hoda
      else if (t ≤ 2T/4 i t > T/4) then
        Primeniti strategiju hoda na stomaku
      else if (t ≤ 3T/4 i t > 2T/4) then
        Primeniti strategiju koordinacije u lovu
      else
        Primeniti strategiju kooperacije u lovu
      end if
    end for
  end for
  t = t + 1
end while
Vratiti najbolje rešenje
```

7.2 Predloženi modifikovani pristup RSA algoritmu

Prioritet eksploatacije nad eksploracijom u RSA algoritmu dovodi do nedostatka raznovrsnosti među populacijom i rane konvergencije. To implicira da početne pozicije rešenja značajno utiču na konačne rezultate. Cilj ovog istraživanja je poboljšati RSA algoritam rešavajući problem ograničenog istraživanja obezbeđivanjem dovoljne raznolikosti populacije tokom inicijalizacije i tokom izvršenja. Da bi se to postiglo, sprovedena su dva unapređenja osnovne metaheuristike RSA: novi pristup inicijalizaciji i mehanizam za očuvanje raznovrsnih rešenja tokom izvršenja algoritma.

7.2.1 Nova šema inicijalizacije

Metod koji je predstavljen u ovom istraživanju koristi tradicionalnu jednačinu inicijalizacije kako bi generisao skup jedinki u početnoj populaciji:

$$X_{i,j} = lb_j + \psi \cdot (ub_j - lb_j), \tag{72}$$

gde $X_{i,j}$ označava j-ti element i-tog rešenja, lb_j i ub_j predstavljaju donje i gornje granice komponente j, dok je ψ je proizvoljna vrednost dobijena iz normalne raspodele unutar granica $[0, 1]$.

Ipak, studija [99] je pokazala da uključivanje pristupa zasnovanog na kvazi-refleksiji (eng. quasi-reflection-based learning - QRL) [99] u populaciju koja je dobijena uz pomoć jednačine (72), može omogućiti istraživanje šireg prostora pretrage. Kao rezultat toga, za svaku komponentu j koja pripada rešenju (X_j), kvazi-refleksivna-suprotna komponenta (eng. quasi-reflective-opposite component) (X_j^{qr}) se generiše na sledeći način:

$$X_j^{qr} = \text{rnd}\left(\frac{lb_j + ub_j}{2}, x_j\right), \tag{73}$$

gde *rnd* omogućava izbor proizvoljnog broja unutar granica $\left[\frac{lb_j + ub_j}{2}, x_j\right]$.

Prema QRL postupku, predloženi pristup inicijalizaciji ne utiče na složenost algoritma u pogledu broja funkcija evaluacije (eng. Function Evaluation Frequencies FFEs), jer generiše samo polovinu celokupne populacije $(NP/2)$. Postupak inicijalizacije koji se koristi u ovom istraživanju je prikazan u Algoritmu 8.

Algorithm 8 Predložena procedura inicijalizacije koja uključuje QRL metod

Korak 1: Generisanje početne populaciju P^{init} od $NP/2$ jedinki pomoću jednačine: (72).
Korak 2: Formiranje QRL populacije P^{qr} na osnovu P^{init} prema:(73)
Korak 3: Formiranje ukupne početne populacije P spajanjem P^{init} i P^{qr}: $(P \cup P^{qr})$
Korak 4: Izračunavanje fitnesa svake jedinke koja pripada populaciji P.
Korak 5: Rangiranje agenata iz populacije P prema rezultatima njihove evaluacije.

Intenzivna ispitivanja su pokazala da ovaj pristup inicijalizaciji pokazuje dve važne prednosti. Prva prednost je povećanje raznovrsnosti početne populacije, što može poboljšati rezultate algoritma na početku izvršavanja. Druga, omogućava algoritmu da pokrije širi prostor pretrage sa istom veličinom populacije, što doprinosi početnom ubrzanju postupka pretrage.

7.2.2 Procedura za očuvanje raznovrsnosti populacije

Da bi se procenilo da li mehanizam pretrage algoritma konvergira ili divergira, jedna od metoda je procena raznovrsnosti (diverziteta) populacije, što je objašnjeno

u [27]. Ova studija koristi novu definiciju merenja raznovrsnosti populacije, posebno korišćenjem $L1$ norme. Ova norma uzima u obzir raznovrsnosti koje proizlaze iz dva faktora: rešenja generisana od strane populacije i dimensionalnosti problema.

Takođe, [27] naglašavaju važnost podataka dobijenih iz dimenzionalnog elementa $L1$ norme, koji se može koristiti za evaluaciju mehanizma pretrage proučavanog algoritma.

Pretpostavimo da m predstavlja broj rešenja u populaciji, a n označava dimenzionalnost problema. $L1$ norma se može izračunati kako je prikazano u jednačinama (74) do (76):

$$\overline{x_j} = \frac{1}{m} \sum_{i=1}^{m} x_{ij} \tag{74}$$

$$\Theta_j^p = \frac{1}{m} \sum_{i=1}^{m} \left| x_{ij} - \overline{x}_j \right| \tag{75}$$

$$\Theta^p = \frac{1}{n} \sum_{i=1}^{n} \Theta_j^p, \tag{76}$$

U ovom kontekstu, $\overline{x}$ se odnosi na niz koji sadrži prosečne pozicije rešenja duž svake dimenzije, dok Θ_j^p predstavlja niz raznovrsnosti pozicija jedinki, izračunatih pomoću $L1$ norme. Θ^p označava ukupnu raznovrsnost populacije kao skalarnu vrednost.

Tokom početnih rundi izvršavanja algoritma, raznovrsnost populacije trebalo bi da bude visoka jer se rešenja generišu korišćenjem standardne jednačine za inicijalizaciju (72). Međutim, dok metoda konvergira ka optimalnom ili suboptimalnom rešenju u kasnijim izvršavanjima algoritma, raznovrsnost bi trebalo dinamički da se smanjuje. Kako bi se rešio ovaj problem, poboljšani RSA algoritam predložen u ovoj studiji koristi $L1$ normu za regulisanje raznovrsnost populacije tokom celokupnog izvršavanja. To se postiže pomoću dinamičkog kontrolnog parametra *praga raznovrsnosti*, koji je predstavljen kao Θ_t.

Predložena je tehnika za očuvanje raznovrsnosti unutar populacije uvođenjem dodatnog kontrolnog faktora, nazvanog nrs, koji određuje broj jedinki koje treba zameniti. Pristup funkcioniše na sledeći način: na početku algoritma, postavlja se dinamički prag raznovrsnosti, označen kao Θ_{t0}. Tokom svake runde izvršavanja, procenjuje se i upoređuje najnovija raznovrsnost populacije, predstavljena kao Θ^P,

sa dinamičkim pragom raznovrsnosti, Θ_t. Ako je uslov $\Theta^P < \Theta_t$ ispunjen, što ukazuje da je raznovrsnost populacije nedovoljna, najlošijih nrs jedinki se zamenjuje slučajnim rešenjima koja se stvaraju korišćenjem metode slične onoj korišćenoj za inicijalizaciju populacije.

Nakon sprovođenja empirijskih simulacija i teorijske analize, formula za izračunavanje Θ_{t0} može se izraziti na sledeći način.

$$\Theta_{t0} = \sum_{j=1}^{NP} \frac{(ub_j - lb_j)}{2 \cdot NP} \tag{77}$$

Kako algoritam napreduje, očekuje se da će se populacija postepeno približavati optimalnom području pretrage. Zbog toga se dinamički prag raznolikosti, Θ_t, mora smanjivati u odnosu na svoju početnu vrednost, Θ_{t0}, koja se izračunava primenom jednačine (77). Da bi se postiglo smanjenje Θ_t, može se koristiti linearna opadajuća funkcija, kako je prikazano u jednačini (78), pri čemu T označava maksimalan broj rundi, a Θ_{t0} predstavlja početni prag raznovrsnosti.

$$\Theta_{t+1} = \Theta_t - \Theta_t \cdot \frac{t}{T}, \tag{78}$$

Ovde, t i $t+1$ predstavljaju trenutnu i narednu rundu. T takođe označava maksimalan broj iteracija. Kako algoritam napreduje, dolazi do dinamičkog smanjenja Θ_t, i na kraju, opisani mehanizam više neće biti korišćen, bez obzira na Θ^P.

7.2.3 Struktura predloženog algoritma

Pošto je predstavljeni modifikovani RSA algoritam, poboljšao izvanredne performanse osnovnog RSA, nazvan je unapređeni RSA, odnosno ERSA, a njegova struktura prikazana je pomoću Algoritma 9. Proučavajući predloženi pseudo-kod, moguće je primetiti da su predložene modifikacije integrisane u osnovnu varijantu RSA koja je data Algoritmom 7 [2].

7.3 Dekompozicija signala

Da bi se kompleksni signali ulaznih karakteristika podelili na niz jednostavnijih podsiganala koji se mogu lakše obrađivati, koristi se proces dekompozicije signala. U ovoj disertaciji korišćene su dve tehnike VMD [36] i EMD [57], koje će biti bliže

Algorithm 9 ERSA Pseudo-kod

Kreiranje početnog skupa rešenja (X) koristeći Algoritam 8
Izračunavanje vrednosti Θ_{t0} i Θ_t
while $t <$ maksimalan broj iteracija **do**
  Procena fitnesa agenta
  Sačuvati najboljeg prethodno pronađenog agenta (P=X*)
  Ažurirati vrednosti η, R, P prema jednačinama (66), (67) i (69)
  Ažuriranje lokacije rešenja korišćenjem procedure date u Algoritmu 7
  Izračunavanje Θ^P
  if $(\Theta^P < \Theta_t)$ **then**
    Zamena najslabije nrs jedinke agentima dobijenim pomoću (72)
  end if
  Ažuriranje Θ_t
end while
Return najboljeg pronađenog agenta kao globalni optimum
Return opšte i detaljne statističke rezultate i vizualizaciju

objašnjene.

7.3.1 Varijaciona dekompozicija

Varijaciona dekompozicija (eng. Variational Mode Decomposition - VMD)[102] je metodologija za dekompoziciju signala na osnovne komponente. Osnova ovog metoda je Vinerovo filtriranje (eng. Wiener filtering) i Hilbertova transformacija (eng. Hilbert transform) [134]. Ovaj adaptivni metod dekompozicije signala može da razloži dati signal $f(t)$ na nekoliko komponentnih signala $u_k(t)$ unutar ograničenja propusnog opsega oko centralne frekvencije ω_k prema jednačini (79)[126].

$$\min_{\{u_k\},\{\omega_k\}} \left\{ \sum_{k=1}^{K} \left\| \partial_t \left[\left(\delta(t) + \frac{j}{\pi t} \right) * u_k(t) \right] e^{-j\omega_k t} \right\|_2^2 \right\}, s.t. \sum_{k=1}^{K} u_k = f(t) \tag{79}$$

gde K predstavlja broj dekomponiranih modova, $u_k = u_1, u_2, \ldots, u_k$ su modalne komponente s centralnim frekvencijama $\omega_k = \omega_1, \omega_2, \ldots, \omega_k$, ∂_t označava parcijalnu derivaciju. $\delta(t)$ predstavlja Diracovu raspodelu, $f(t)$ prikazuje originalni ulazni signal, $u_k(t)$ predstavlja k-ti podniz od $f(t)$, a * označava operator konvolucije.

Lagranžov multiplikatorski operator λ i kvadratni faktor kazne (eng. quadratic penalty factor) α uključeni su kako bi se unapredilo optimalno rešenje ograničene varijacije na sledeći način:

$$L(\{u_k\},\{\omega_k\},\lambda) = \alpha \sum_{k=1}^{K} \left\| \partial_t \left[\left(\delta(t) + \frac{j}{\pi t} \right) * u_k(t) \right] e^{-j\omega_k t} \right\|_2^2 + \left\| f(t) - \sum_{k=1}^{K} u_k(t) \right\|_2^2 + \left\langle \lambda(t), f(t) - \sum_{k=1}^{K} u_k(t) \right\rangle$$

Lagranžova funkcija se transformiše iz vremenskog domena u frekvencijski domen. Metod alternativnih pravaca multiplikatora (eng. Alternate Direction Method of Multipliers - ADMM) koristi se kako bi se minimizirao optimizacijski problem. Modalne komponente u_k i njihove centralne frekvencije ω_k izračunavaju se pomoću sledećih jednačina, redom: jednačina (80) i jednačina (81).

$$\hat{u}_k^{n+1}(\omega) = \frac{\hat{f}(\omega) - \sum_{i \neq k} \hat{u}_i(\omega) + \frac{\hat{\lambda}(\omega)}{2}}{1 + 2\alpha(\omega - \omega_k)^2} \tag{80}$$

$$\omega_k^{n+1} = \frac{\int_0^\infty \omega \left| \hat{u}_k(\omega) \right|^2 d\omega}{\int_0^\infty \left| \hat{u}_k(\omega) \right|^2 d\omega} \tag{81}$$

gde je n je broj iteracija, a λ Lagranžov operator dat jednačinom (82).

$$\hat{\lambda}^{n+1}(\omega) = \hat{\lambda}^n(\omega) + \tau \left[\hat{f}(\omega) - \sum_{k=1}^{K} \hat{u}_k^{n+1}(\omega) \right] \tag{82}$$

Iterativni proces će se izvršavati sve dok uslov dat jednačinom Eq. (83) ne bude ispunjen.

$$\frac{\sum_{k=1}^{K} \left\| \hat{u}_k^{n+1}(\omega) - \hat{u}_k^n(\omega) \right\|_2^2}{\left\| \hat{u}_k^n(\omega) \right\|_2^2} < \epsilon \tag{83}$$

7.3.2 Empirijska dekompozicija

Empirijska dekompozicija (eng. Empirical Mode Decomposition - EMD)[103] je pristup dekompoziciji signala koji smanjuje količinu šuma u nestacionarnim vremenskim serijama podataka [33]. Koriste se Fourierova i Wavelet analiza. Ova tehnika daje dobre rezultate prilikom analize vremenskih serija podataka o brzini vetra. Korišćenjem EMD algoritma, kompleksni podaci vremenskih serija mogu se de-

komponovati na ograničen broj unutrašnjih modalnih funkcija (eng. Intrinsic Mode Functions - IMFs) [123].

Postupak EMD-a je sledeći:

(1) Određivanje lokalne maksimalne i minimalne vrednosti za bilo koji obrađeni signal $x(t)$. Snimanje (pamćenje) $h_1(t)$ kao razliku između $x(t)$ i srednje vrednosti gornje i donje anvelope $m_1(t)$, prema jednačini (84).

$$h_1(t) = x(t) - m_1(t) \tag{84}$$

(2) Signal $h_1(t)$ koji je izdvojen iz originalnog signala ima tendenciju da sadrži najvišu frekvencijsku komponentu signala. Signal razlike, $r_1(t)$ dobijen je razdvajanjem $h_1(t)$ od $x(t)$. Na ovaj način, visokofrekventna komponenta je uklonjena. Koraci filtriranja se ponavljaju, sa $r_1(t)$ kao novim signalom, s ciljem da rezidualni signal u n-tom koraku postane monotona funkcija. Ovo je prikazano jednačinom (85).

$$\begin{cases} r_1(t) = x(t) - h_1(t) \\ r_2(t) = r_1(t) - h_2(t) \\ \quad \vdots \qquad \vdots \\ r_n(t) = r_{n-1}(t) - h_n(t) \end{cases} \tag{85}$$

u kojoj se $x(t)$ može predstaviti kao suma od n IMF-a i jednog reziduala, prema jednačini (86).

$$x(t) = \sum_{j=1}^{n} h_j(t) + r_n(t) \tag{86}$$

gde je $r_n(t)$ rezidual koji označava prosečan trend signala, $h_j(t)$ je j-ta IMF, pri čemu $j = 1, 2, ..., n$, predstavlja različite komponente signala u smeru od visokih prema niskim frekvencijama.

Standardna devijacija (SD) je data jednačinom (87). Uglavnom se postavlja u rasponu od 0.2 do 0.3.

$$SD = \sum_{j=1}^{n} \frac{\left|h_j(t) - h_{j-1}(t)\right|^2}{\left|h_{j-1}(t)\right|^2} \tag{87}$$

7.4 Metrike performansi generisanih modela

Simulacioni rezultati posmatranih modela validirani su primenom skupa tradicionalnih mera mašinskog učenja, tačnije srednje kvadratne greške (eng. Mean squared error - MSE) koja se računa prema jednačini (88), korena srednje kvadratne greške (eng. Root mean square error - RMSE) koja se izračunava prema jednačini (89), srednje apsolutne greške (Mean absolute error - MAE) određene jednačinom (91) i koeficijenta determinacije (eng. Coefficient of determination - R2), koji se može odrediti pomoću jednačine (91).

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{p}_i - p_i\right)^2 \tag{88}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{p}_i - p_i\right)^2} \tag{89}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{p}_i - p_i\right| \tag{90}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(p_i - \hat{p}_i\right)^2}{\sum_{i=1}^{n}\left(p_i - \bar{p}\right)^2}, \tag{91}$$

gde p_i i $\hat{p}_i$ označavaju nizove koji sadrže posmatrane i predviđene vrednosti, oba sa N elemenata. U ovoj disertaciji MSE je korišćena kao funkcija cilja koju je potrebno minimizovati.

7.5 SHAP analiza

SHAP (eng. SHapley Additive exPlanations) [79] predstavlja metode za tumačenje rezultata modela mašinskog učenja, posebno onih koji su *crna kutija* ili teški za tumačenje. Metodologija SHAP-a se zasniva na računanju *Shapley vrednosti*, koncepta iz teorije kooperativnih igara, kako bi se objasnila uloga dostupne karakteristike i njen uticaj na odluke. Shapley vrednost se određuje za određenu karakteristiku, koja predstavlja prosečan uticaj te karakteristike na izlaz modela preko svih mogućih podskupova karakteristika.

Da bi se izračunala Shapley vrednost za određenu karakteristiku, prvo se definiše referentna vrednost za tu karakteristiku. Ova referentna vrednost može biti prosečna

vrednost te karakteristike u skupu podataka, ili može biti vrednost koju korisnik definiše. Zatim se kreira skup svih mogućih podskupova karakteristika koji sadrže karakteristiku koja nas zanima. Na primer, ako nas zanima Shapley vrednost za karakteristiku A, možemo kreirati podskupove koji uključuju samo A, ili A i B, ili A i C, i tako dalje.

Za svaki podskup računamo doprinos karakteristike izlazu modela u odnosu na referentnu vrednost. Ovaj doprinos može biti pozitivan ili negativan, uzrokujući povećanje ili smanjenje vrednosti karakteristike u podskupu na izlazu modela. Zatim računamo prosečan doprinos karakteristike preko svih mogućih podskupova, što nam daje Shapley vrednost.

Matematički, Shapley vrednost za karakteristiku j se određuje pomoću sledeće jednačine:

$$\phi_j = \sum_{S \subseteq M \setminus j} \frac{|S|!(|M| - |S| - 1)!}{|M|!}(f(S \cup j) - f(S)) \tag{92}$$

Ovde, M predstavlja skup svih karakteristika, a $f(S)$ je izlaz modela za dati podskup karakteristika S. Izraz unutar sume izračunava marginalni doprinos karakteristike j podskupu S, a suma izračunava prosečan doprinos preko svih mogućih podskupova.

U praksi, možemo procenjivati Shapley vrednosti za model korišćenjem tehnike nazvane *Kernel SHAP*. Ovo uključuje generisanje skupa 'pozadinskih' instanci koje su reprezentativne za skup podataka, a zatim korišćenje tih instanci kako bismo procenili očekivani izlaz modela za svaki podskup karakteristika. Shapley vrednosti se zatim mogu izračunati koristeći ove očekivane izlaze.

Kada dobijemo Shapley vrednosti za sve karakteristike, možemo ih koristiti kako bismo kreirali SHAP grafički prikaz, koji prikazuje doprinos svake karakteristike izlazu modela. Ovaj prikaz može pomoći u identifikaciji koje karakteristike su najvažnije za predviđanja modela, i kako svaka karakteristika doprinosi tim predviđanjima.

7.6 Eksperimenti sa standardnim funkcijama sa ograničenjima opsega

Pre ocene performansi ERSA metaheuristika na praktičnom optizovanju rekurentnih neuronskih mreža za predviđanje vremenskih serija energije vetra, predložena

metoda prvo je testirana na standardnim benčmark funkcijama sa ograničenjima opsega (neograničenim funkcijama).

Test skup CEC2019 je odabran iz dva razloga: ova grupa funkcija je izazovnija i kompleksnija od drugih skupova za testiranje (na primer, standardni test primeri i CEC2017); osnovni RSA je takođe bio evaluiran na njima kada je ovaj pristup prvi put uveden [2]. Ovaj paket sastoji se od deset funkcija, a detalji o njima (ime, dimenzija, ograničenja pretrage, globalni optimum) su prikazani u Tabeli 2.

Simulacije CEC2019 su sprovedene kako sa predstavljenim ERSA, tako i sa originalnim RSA algoritmom. Dodatno, kako bi se uporedna analiza proširila, u razmatranje su uzete i druge metaheuristike najnovije generacije: optimizacija rojem čestica (eng. Particle Swarm Optimization - PSO)[70], veštačka kolonija pčela (eng. Artificial Bee Colony - ABC)[68], algoritam svitaca (eng. Firefly Algorithm - FA)[130], algoritam Harisovog jastreba (eng. Harris Hawks Optimization - HHO)[54], algoritam kitova (eng. Whale Optimization Algorithm - WOA)[90] i algoritam šimpanzi (eng. Chimp Swarm Optimization - ChOA)[71]. Ovakav skup algoritama je izabran kako bi se postigao balans između tradicionalnih metoda poput PSO i novijih, kao što je ChOA.

U istraživanjima u okviru ove disertacije, definisani su slični eksperimentalni uslovi kao u [2], tako da su maksimalni broj iteracija (T) i broj agenata u populaciji (N) podešeni na 500 i 30, respektivno. Dodatno se, zbog stohastičkog ponašanja metoda, eksperimenti ponavljaju nezavisno 30 puta ($runtime = 30$), sa zabeleženim najboljim (Best), najgorim (Worst), srednjim (Average) vrednostima i standardnom devijacijom (STD).

Sve evaluirane metode su implementirane specifično za ovu disertaciju, a rezultati dobijeni u [2] nisu uzeti u obzir. Međutim, treba napomenuti da su veoma slični rezultati za osnovni RSA, WOA i PSO kao oni prijavljeni u [2] dobijeni u simulacijama sprovedenim u okviru ovog istraživanja.

Tabela 2: *Pregled CEC2019 benčmark funkcija.*

Br.	Funkcije	$F_i^* = F_i(x^*)$	Dim.	Oblast pretrage
1	Storn's Chebyshev Polynomial Fitting Problem	1	9	[-8192, 8192]
2	Inverse Hilbert Matrix Problem	1	16	[-16384, 16384]
3	Lennard-Jones Minimum Energy Cluster	1	18	[-4,4]
4	Rastrigin's Funkcija	1	10	[-100,100]
5	Griewangk's Funkcija	1	10	[-100,100]
6	Weierstrass Funkcija	1	10	[-100,100]
7	Modified Schwefel's Funkcija	1	10	[-100,100]
8	Expanded Schaffer's F6 Funkcija	1	10	[-100,100]
9	Happy Cat Funkcija	1	10	[-100,100]
10	Ackley Funkcija	1	10	[-100,100]

Poređenje mera performansi posmatranih algoritama dato je u Tabeli 3, gde su najbolji postignuti rezultati označeni boldiranim slovima.

Rezultati testiranja CEC2019 funkcija pokazuju da je predloženi ERSA u proseku pokazao izvanredne performanse. Takođe je očigledno, da ovo nije slučaj za sve test funkcije. Prema NFL teoremi optimizacije, očekivano je da ni jedan pojedinačni algoritam nije podjednako efikasan u svim primenama. Zbog toga je potrebno sprovesti eksperimente kako bi se utvrdilo koji algoritmi najbolje odgovaraju određenom problemu.

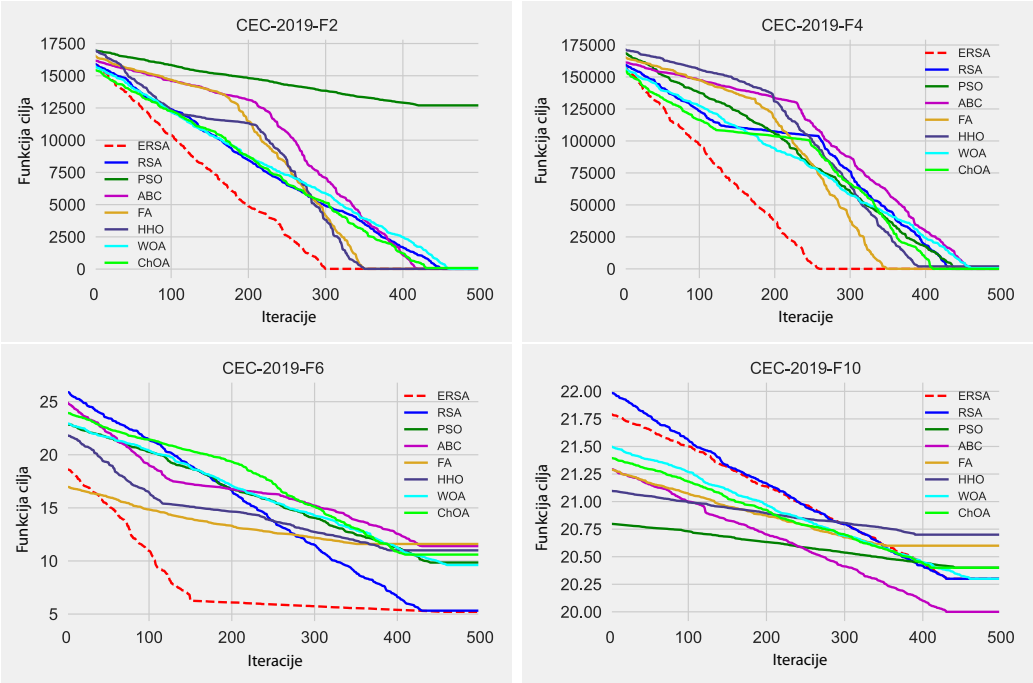
Dodatne analize i observacije pomažu u razumevanju prednosti i slabosti predstavljenih algoritama u poređenju sa savremenim metodama. Interesantno je da u određenim funkcijama, poput $F9$, iako algoritam nije pokazao najbolje performanse, ipak je postigao najbolji rezultat STD, što ukazuje da, iako predloženi algoritam nije optimalno rešenje (za malu vrednost), on je najstabilnija i najpouzdanija opcija. Ova opservacija se donekle odnosi i na funkciju $F4$, kod koje i pored dobrog rezultata za najgore i za prosečno izvršavanje, algoritam ne pokazuje najveću pouzdanost, što dodatno naglašava važnost opsežnih eksperimenata i NFL teoreme.

Tabela 3: *Rezultati RSA dobijeni korišćenjem CEC2019 test funkcija.*

Funkcija		ERSA	RSA	PSO	ABC	FA	HHO	WOA	ChOA
CEC	Najgori	**4.38E+04**	6.33E+04	2.23E+13	4.66E+10	3.15E+08	1.15E+10	1.22E+11	1.01E+09
2019	Prosečan	**3.71E+04**	5.65E+04	8.12E+12	2.41E+10	1.01E+08	4.35E+09	5.81E+10	5.12E+08
Func 1	Najbolji	**3.22E+04**	5.06E+04	8.26E+11	4.33E+09	5.03E+05	1.29E+09	1.55E+09	2.57E+08
	STD	**4.65E+03**	4.93E+03	8.60E+12	2.16E+10	1.45E+08	4.19E+09	4.73E+10	2.76E+08
	Rang	**1**	2	8	7	3	5	6	4
CEC	Najgori	1.73E+01	1.73E+01	1.63E+04	1.73E+01	1.73E+01	1.75E+01	1.74E+01	1.35E+02
2019	Prosečan	1.73E+01	1.73E+01	1.27E+04	1.73E+01	1.73E+01	1.75E+01	1.74E+01	8.34E+01
Func 2	Najbolji	1.73E+01	1.73E+01	8.32E+03	1.73E+01	1.73E+01	1.74E+01	1.73E+01	4.35E+01
	STD	**4.56E-08**	1.26E-07	3.13E+03	2.41E-04	3.45E-05	4.68E-02	6.19E-03	3.06E+01
	Rang	1	1	5	1	1	3	2	4
CEC	Najgori	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01
2019	Prosečan	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01
Func 3	Najbolji	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01	1.27E+01
	STD	0.00E+00	5.43E-05	0.00E+00	**5.64E-17**	4.29E-06	3.42E-05	7.43E-07	5.23E-05
	Rang	1	1	1	1	1	1	1	1
CEC	Najgori	**1.32E+01**	2.71E+01	2.36E+01	1.15E+02	1.65E+02	3.61E+03	8.64E+02	2.72E+01
2019	Prosečan	**9.13E+00**	2.45E+01	1.73E+01	5.63E+01	7.54E+01	1.95E+03	5.86E+02	2.43E+01
Func 4	Najbolji	5.02E+00	2.13E+01	**4.96E+00**	1.11E+01	4.88E+01	1.08E+03	2.25E+02	2.07E+01
	STD	3.41E+00	2.44E+00	6.53E+00	3.73E+01	5.13E+01	9.55E+02	2.63E+02	**2.40E+00**
	Rang	**1**	4	2	5	6	8	7	3
CEC	Najgori	1.11E+00	1.09E+00	1.66E+00	1.09E+00	2.68E+00	2.68E+00	1.55E+00	1.71E+00
2019	Prosečan	1.06E+00	1.05E+00	1.45E+00	1.05E+00	2.41E+00	2.37E+00	1.26E+00	1.54E+00
Func 5	Najbolji	1.00E+00	1.00E+00	1.25E+00	1.00E+00	2.17E+00	2.17E+00	1.12E+00	1.31E+00
	STD	5.79E-02	4.66E-02	1.63E-01	4.23E-02	2.57E-01	1.99E-01	**1.59E-01**	1.66E-01
	Rang	2	1	4	1	7	6	3	5
CEC	Najgori	8.76E+00	**8.54E+00**	1.25E+01	1.21E+01	1.21E+01	1.16E+01	1.13E+01	1.14E+01
2019	Prosečan	**5.21E+00**	5.32E+00	9.85E+00	1.14E+01	1.16E+01	1.10E+01	9.62E+00	1.06E+01
Func 6	Najbolji	**2.94E+00**	3.05E+00	8.96E+00	1.05E+01	1.14E+01	9.69E+00	8.27E+00	9.15E+00
	STD	2.66E-01	**2.31E-01**	9.48E-01	4.72E-01	2.38E-01	7.46E-01	1.12E+00	8.05E-01
	Rang	**1**	2	4	7	8	6	3	5
CEC	Najgori	**1.44E+02**	6.51E+02	4.13E+02	1.41E+03	6.63E+02	1.05E+03	1.15E+03	6.06E+02
2019	Prosečan	**4.69E+01**	4.48E+02	2.88E+02	6.55E+02	3.55E+02	8.52E+02	8.13E+02	2.57E+02
Func 7	Najbolji	**8.84E+01**	2.79E+02	3.43E+02	1.85E+02	1.26E+02	4.79E+02	4.64E+02	2.91E+01
	STD	**9.73E+01**	1.53E+02	1.73E+02	4.45E+02	2.31E+02	2.26E+02	2.54E+02	2.93E+02
	Rang	**1**	5	2	6	4	8	7	3
CEC	Najgori	**4.92E+00**	6.55E+00	5.65E+00	6.11E+00	6.85E+00	6.66E+00	6.64E+00	6.63E+00
2019	Prosečan	**4.10E+00**	5.62E+00	4.96E+00	5.71E+00	5.49E+00	5.88E+00	6.11E+00	5.87E+00
Func 8	Najbolji	**2.55E+00**	4.66E+00	4.39E+00	5.13E+00	2.92E+00	5.35E+00	5.65E+00	4.61E+00
	STD	9.24E-01	7.41E-01	6.37E-01	4.76E-01	1.76E+00	4.642-01	**4.24E-01**	8.13E-01
	Rang	**1**	4	2	5	3	7	8	6
CEC	Najgori	2.47E+00	2.45E+00	2.44E+00	2.86E+00	7.16E+00	4.55E+02	6.42E+00	**2.43E+00**
2019	Prosečan	2.46E+00	2.41E+00	2.42E+00	2.62E+00	4.95E+00	2.05E+02	5.43E+00	**2.36E+00**
Func 9	Najbolji	2.45E+00	2.36E+00	2.38E+00	2.43E+00	3.33E+00	4.54E+01	4.06E+00	**2.35E+00**
	STD	**7.39E-07**	3.17E-02	3.41E-02	1.76E-01	1.52E+00	1.55E+02	9.59E-01	1.09E-02
	Rang	4	2	3	5	6	8	7	**1**
CEC	Najgori	2.06E+01	2.04E+01	2.07E+01	**2.01E+01**	2.07E+01	2.08E+01	2.05E+01	2.05E+01
2019	Prosečan	2.03E+01	2.03E+01	2.04E+01	**2.00E+01**	2.06E+01	2.07E+01	2.03E+01	2.04E+01
Func 10	Najbolji	2.01E+01	2.02E+01	2.01E+01	**2.00E+01**	2.05E+01	2.06E+01	2.02E+01	2.04E+01
	STD	3.45E-01	1.36E-01	1.85E-01	**3.97E-02**	4.72E-02	6.64E-02	1.09E-01	5.43E-02
	Rang	2	2	3	1	4	5	2	3
Srednji	rang	**1.5**	2.4	3.4	3.9	4.3	5.7	4.6	3.5
Finalno	rangiranje	**1**	2	3	5	6	8	7	4

Direktno poboljšanje može se primetiti kod $F2$, gde, iako su originalni RSA i unapređeni ERSA postigli identične rezultate za najbolje, prosečne i najgore izvršavanje, predloženi algoritam je ostvario značajno poboljšanje u pogledu stabilnosti, što je pokazao smanjenjem standardne devijacije. Kod F10, oba algoritma su pokazala gore performanse u poređenju sa ABC algoritmom, pri čemu je originalni RSA bio nešto bolji u najgorem izvršavanju, predloženi ERSA postigao bolje rezultate u najboljem izvršavanju, dok im je rezultat za prosečno izvršavanje jednak. Za test funkciju $F3$, većina metaheuristika je pokazala slično dobre rezultate. U većini drugih slučajeva, predloženi algoritam je ostvario poboljšanje performansi u odnosu na originalni i takođe nadmašio konkurentske metaheuristike u tim slučajevima.

Konačno, rangiranje metaheuristika sprovedeno je prema prosečnim ocenama performansi, pri čemu su algoritmi sa najboljim performansama dobili niže rangove, dok su manje optimalni algoritmi postizali progresivno više rangove. Može se primetiti



Slika 33: *Grafovi brzine konvergencije za neke proizvoljne CEC2019 funkcije*

da je predloženi algoritam ostvario najbolji rang u većini slučajeva, a tesno ga je pratio originalni RSA algoritam. Prikazan je i prosečni rang svih funkcija, gde je predloženi ERSA algoritam ostvario srednji rang od 1.5, a originalni RSA od 2.4. Na osnovu ovog rangiranja može se zaključiti da je predstavljeni ERSA pokazao izuzetno dobre rezultate u poređenju sa savremenim algoritmima, dok je istovremeno unapredio izuzetne performanse, koje pruža RSA. Takođe je važno napomenuti da su sva ispitivanja i algoritmi nezavisno implementirani i testirani. Osim toga, dobijeni rezultati su u skladu sa prethodnim radovima [2] koji su na sličan način ocenili originalni RSA algoritam u poređenju sa nekoliko najsavremenijih algoritama.

Na kraju, radi vizualizacije performansi ocenjenih metoda, prikazani su grafovi brzine konvergencije za proizvoljno odabrane instance za srednje izvršavanje, što je prikazano na Slici 33. U slučajevima test funkcija $F2$, $F4$ i $F6$, ERSA relativno brzo konvergira, međutim, za test $F10$, predložena metoda pokazuje relativno stabilne, ali ne i najbolje performanse.

7.7 Pregled korišćenih skupova podataka

Za procenu performansi LSTM modela za predviđanje multivarijantnih vremenskih serija, korišćena su dva skupa podataka. U ovoj disertaciji, dat je kratak opis

oba skupa podataka. Radi lakšeg praćenja, prvi skup podataka nazvan je *Wind dataset*, a drugi *Wind farm dataset*.

7.7.1 Wind dataset

Ovaj skup satnih podataka o proizvodnji i potrošnji električne energije, kao i meteorološki podaci, potiče iz dva osnovna izvora i dostupan je online na inernet adresi [2]. Prvi izvor obuhvata podatke o proizvodnji i potrošnji električne energije za Španiju. Ove podatke je obezbedio javni portal ENTSOE za Transmission Service Operator (TSO). Njima su obuhvataćeni različiti izvori energije kao što su biomasa, geotermalna energija, vetar, solarna energija i fosilna goriva.

Drugi izvor uključuje relevantne meteorološke podatke za Valensiju u Španiji. Informacije za ovaj deo skupa podataka obezbeđene su putem sajta Weather API dostupnog na web adresi [3]. Ovi podaci obuhvataju satne vrednosti za temperaturu, pritisak, vlažnost vazduha, brzinu vetra, smer vetra i količinu padavina.

Jedinstveni skup podataka obuhvata četvorogodišnje satne vrednosti podataka o vremenu, opterećenju i proizvodnji električne energije za Španiju, za period od 2015. do 2018. godine. Dostupne informacije čine ovaj skup podataka odličnim kandidatom za prognozu proizvodnje električne energije na osnovu dostupnih meteoroloških podataka. U ovoj disertaciji, podaci o proizvodnji električne energije na kopnu koriste se kao ciljna promenljiva, dok su dostupni podaci o vremenu korišćeni kao ulazi.

Međutim, s obzirom na to da je ovo relativno veliki skup podataka, za simulacije je uzet samo period od 1. januara 2018. do 31. decembra 2018. godine i sastoji se od 8759 opservacija. Podela skupa podataka za trening, validaciju i testiranje prikazana na funkciji cilja, može se videti na Slici 34 [94].

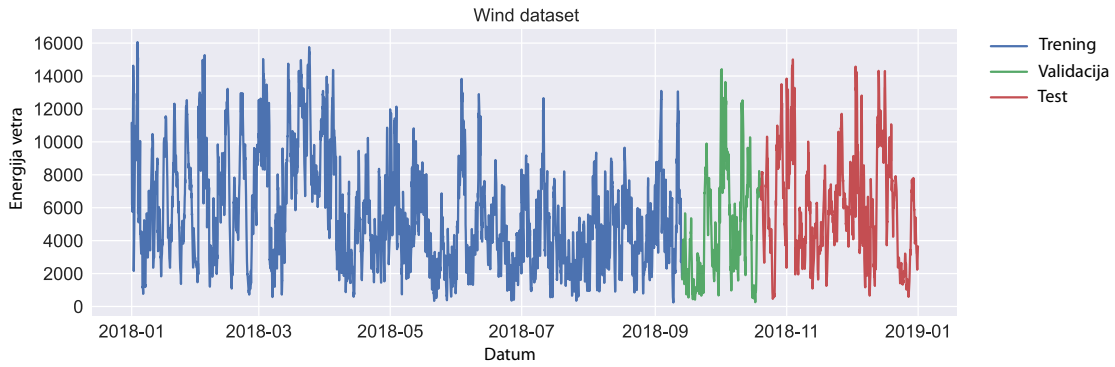
7.7.2 Wind farm dataset

Ovaj dataset ranije je korišćen u okviru GEFCom2012 izazova (GEFCom2012 challenge) [56] i bio dostupan na Kaggle platformi na adresi[4]. Wind farm dataset je dobio novu namenu za potrebe prognoze proizvodnje energije vetra. Ovaj skup

[2] `https://www.kaggle.com/datasets/nicholasjhana/energy-consumption-generation-prices-and-weather?select=energy_dataset.csv`

[3] `https://openweathermap.org/api`

[4] `https://www.kaggle.com/competitions/global-energy-forecasting-competition-2012-load-forecasting/data`



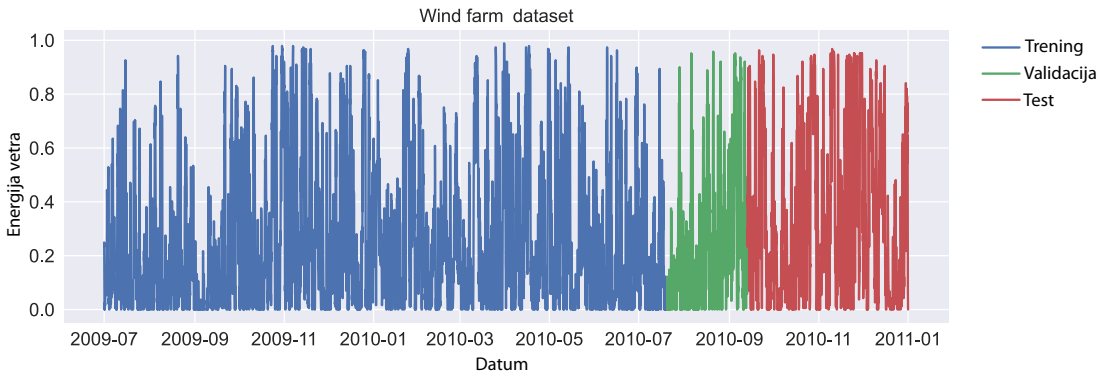
Slika 34: *Wind dataset podeljen na podatke za trening, validaciju i testiranje*

podataka prvobitno je predstavljen sa ciljem unapređenja prakse predviđanja i njegove primene u različitim industrijama, a služi i kao veza između akademskog istraživanja i industrijske prakse. U ovoj disertaciji, ovaj skup podataka se koristi za procenu performansi predloženih modela pri rešavanju kompleksnog zadatka predviđanja proizvodnje energije vetra.

Ovaj skup podataka obuhvata meteorološke i podatke o proizvodnji energije za 7 neimenovanih vetroparkova na kontinentalnom delu Kine. Podaci o proizvodnji električne energije farmi su normalizovani kako bi se obezbedila anonimnost. Podaci u ovom datasetu prate 24-časovne prognoze relevantnih meteoroloških podataka o vetru, kreirane na svakih 12 sati. Brzina vetra i smer vetra dostupni su zajedno sa zonalnim i meridijalnim komponentama vetra za svaki vetropark.

U eksperimentalne svrhe, relevantni meteorološki podaci su svedeni na 12-časovne predikcije u svakoj prognozi, kako bi se dobila satna rezolucija. Zatim su ovi podaci kombinovani sa dostupnim normalizovanim podacima o stvarnoj proizvodnji energije vetra za svaki odgovarajući vetropark na satnom nivou. Originalni skup podataka obuhvata četiri i po godine podataka sa satnom rezolucijom, pri čemu je poslednjih pola godine podataka rezervisano za testiranje. Međutim, podaci za poslednju polovinu godine nisu dostupni.

Zbog ogromne količine opservacija dostupnih u ovom skupu podataka, kao i zbog vrednosti koje nedostaju u kasnijim delovima podataka, tokom izvođenja eksperimenta korišćen je manji deo dostupnih podataka. Velika količina podataka zahteva intenzivno korišćenje resursa prilikom obuke i evaluacije modela. Iz tog razloga se eksperimentalni skup podataka korišćen u simulacijama sastoji od podataka za dve godine (od 1. januara 2009. do 31. decembra 2010), za jedan neimenovani vetropark.



Slika 35: *Wind farm dataset podeljen na podatke za trening, validaciju i testiranje*

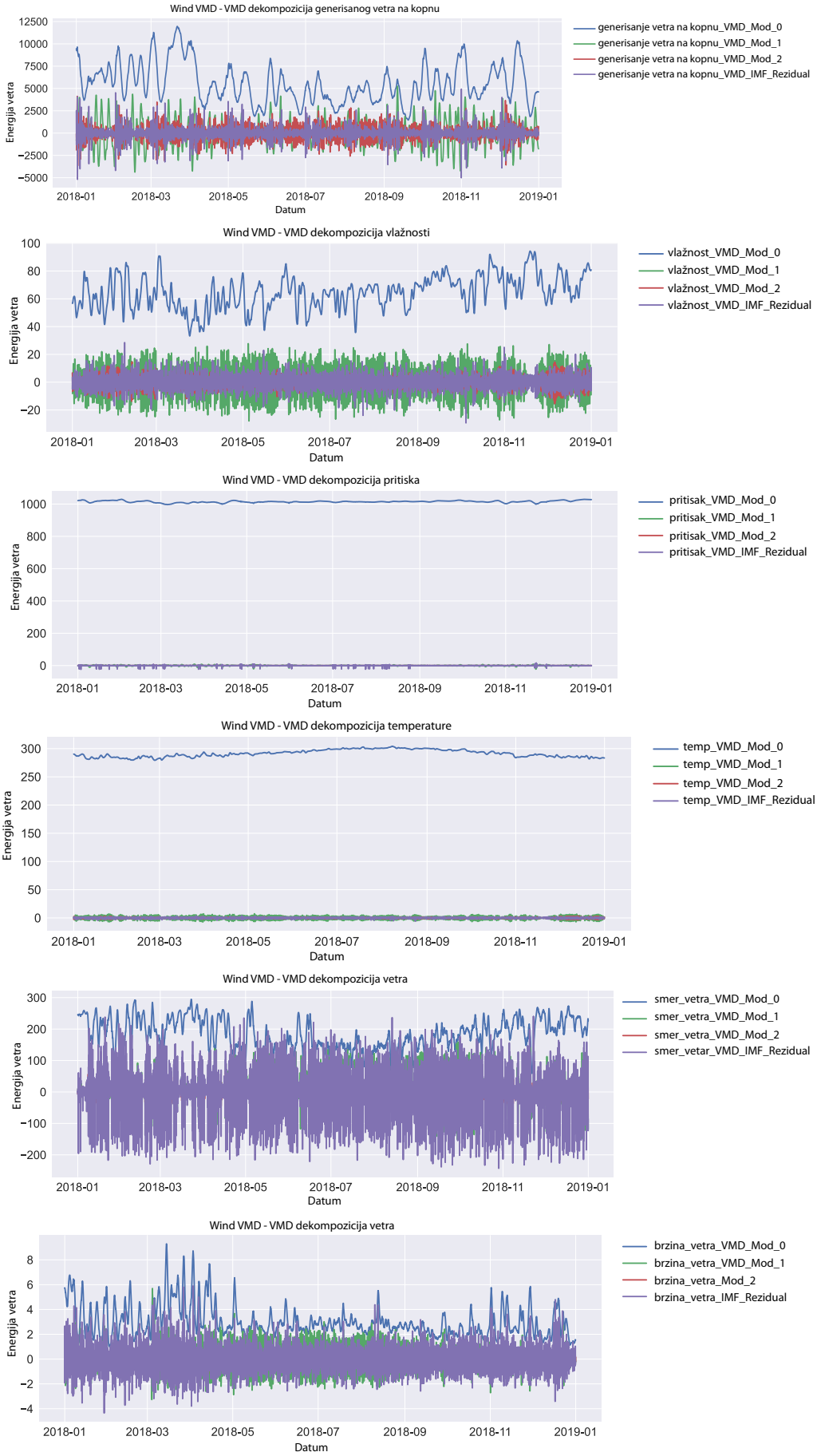
Poslednji deo korišćenog skupa podataka sadrži ukupno 13176 instanci za vetropark 2. Podela skupa podataka za trening, validaciju i testiranje prikazana na funkciji cilja, može se videti na Slici 35.

7.8 Primena tehnika dekompozicije

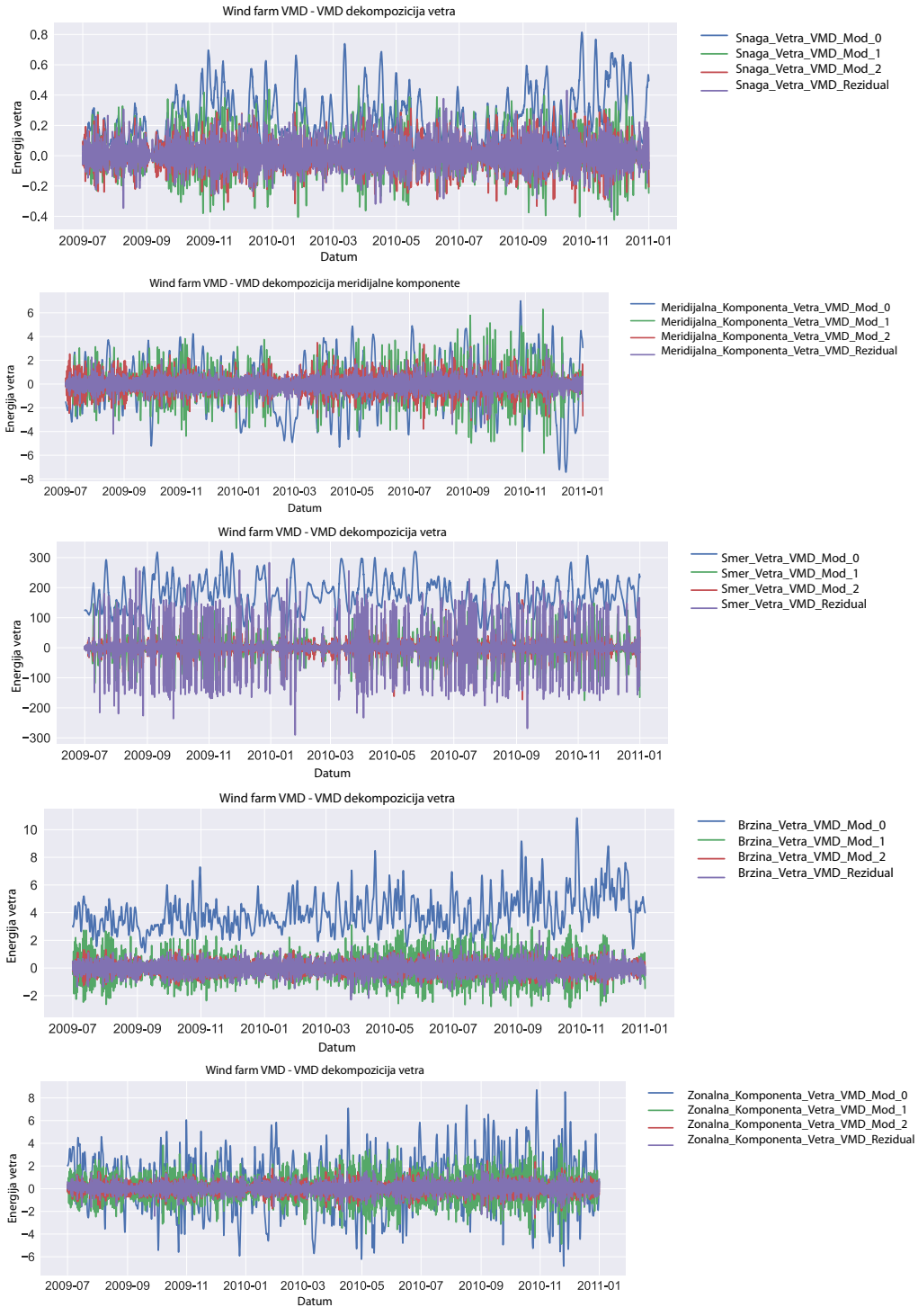
Početna faza eksperimentalnog dela uključuje primenu tehnika dekompozicije na dostupne ulazne karakteristike podataka. Ovo se radi kako bi se kompleksni signali ulaznih karakteristika podelili na niz jednostavnijih podsiganala koji se mogu kasnije koristiti za predviđanje. Međutim, ovaj proces povećava broj karakteristika koje mreža mora da obradi.

Dve tehnike koje su testirane u ovom radu uključuju VMD [36] i EMD [57]. VMD je izvršena sa vrednošću k jednakom 3, što kao rezultat daje ukupno četiri signala. Tri signala predstavljaju dobijene modove, dok je četvrti rezidualna vrednost koja nije obuhvaćena nijednim modom. Svi VMD modovi i reziduali za Wind dataset i Wind farm dataset prikazani su na Slici 36 i Slici 37, respektivno.

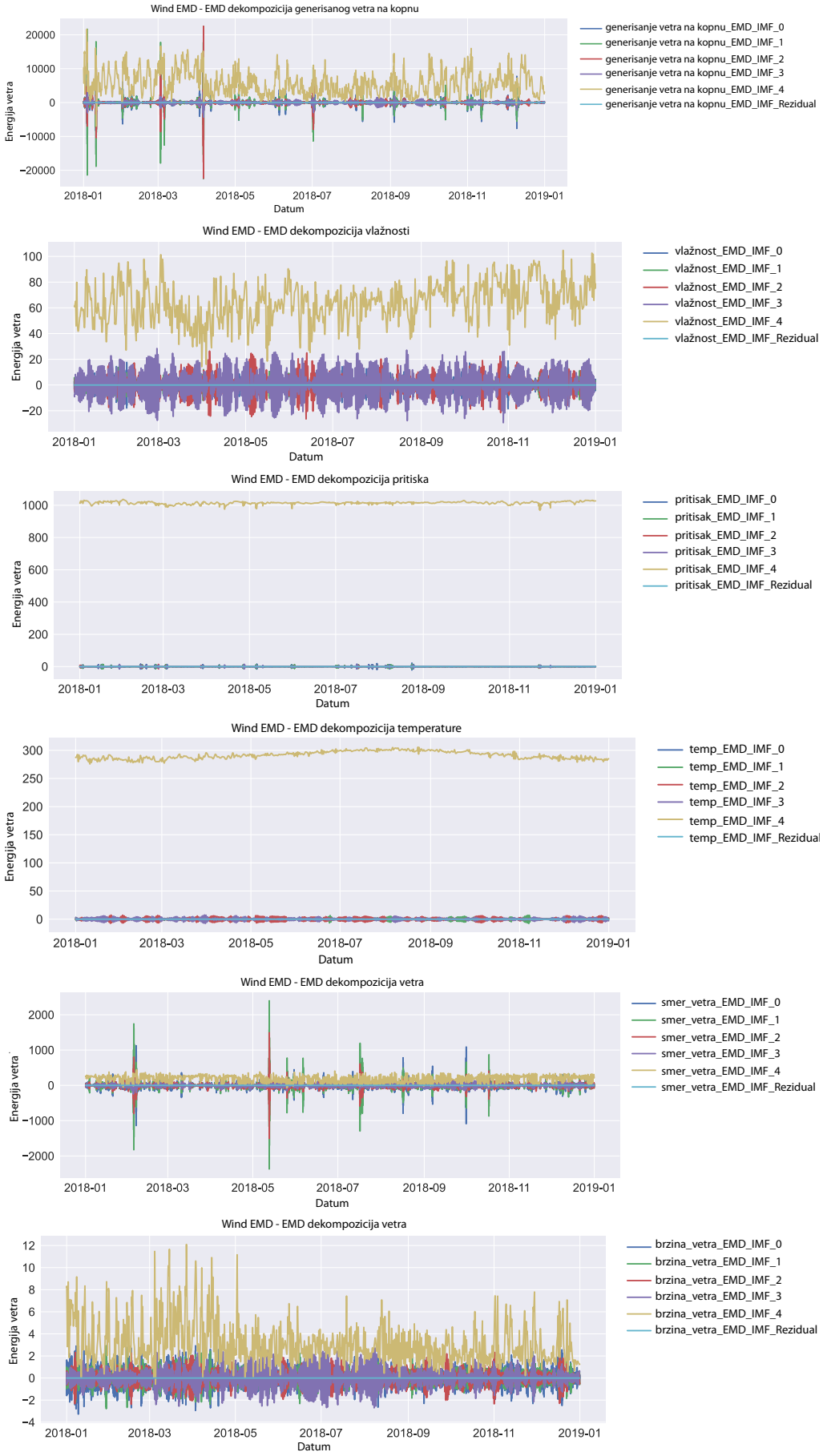
Slično tome, tehnika EMD ograničava broj IMF komponenti na maksimalno 4. Važno je napomenuti da, kada se ne može izdvojiti nova IMF komponenta putem dekompozicije, EMD se prekida ranije, što rezultira manjim brojem odgovarajućih komponenti. Pored IMF komponenti, dodaje se dodatna rezidualna komponenta koja se sastoji od signala koji ne mogu biti dodeljeni nekom IMF-u. Ovo kao rezultat daje maksimalno 5 podsignala po ulaznoj karakteristici. Svi EMD IMF-ovi i reziduali za Wind dataset i Wind farm dataset prikazani su na Slici 38 i Slici 39, redom.



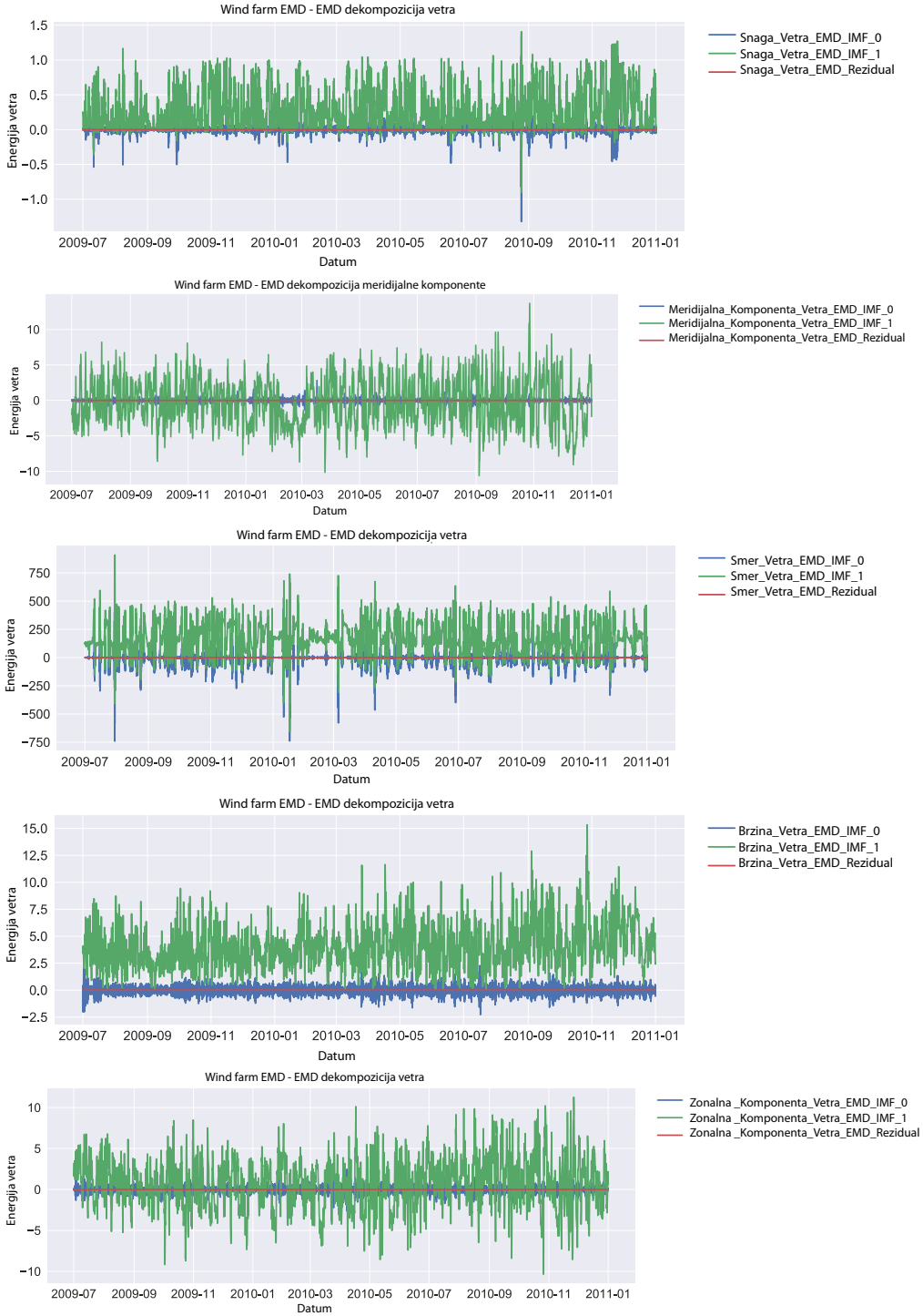
Slika 36: *VMD modovi i reziduali ulaznih karakteristika za Wind dataset*



Slika 37: *VMD modovi i reziduali ulaznih karakteristika za Wind farm dataset*



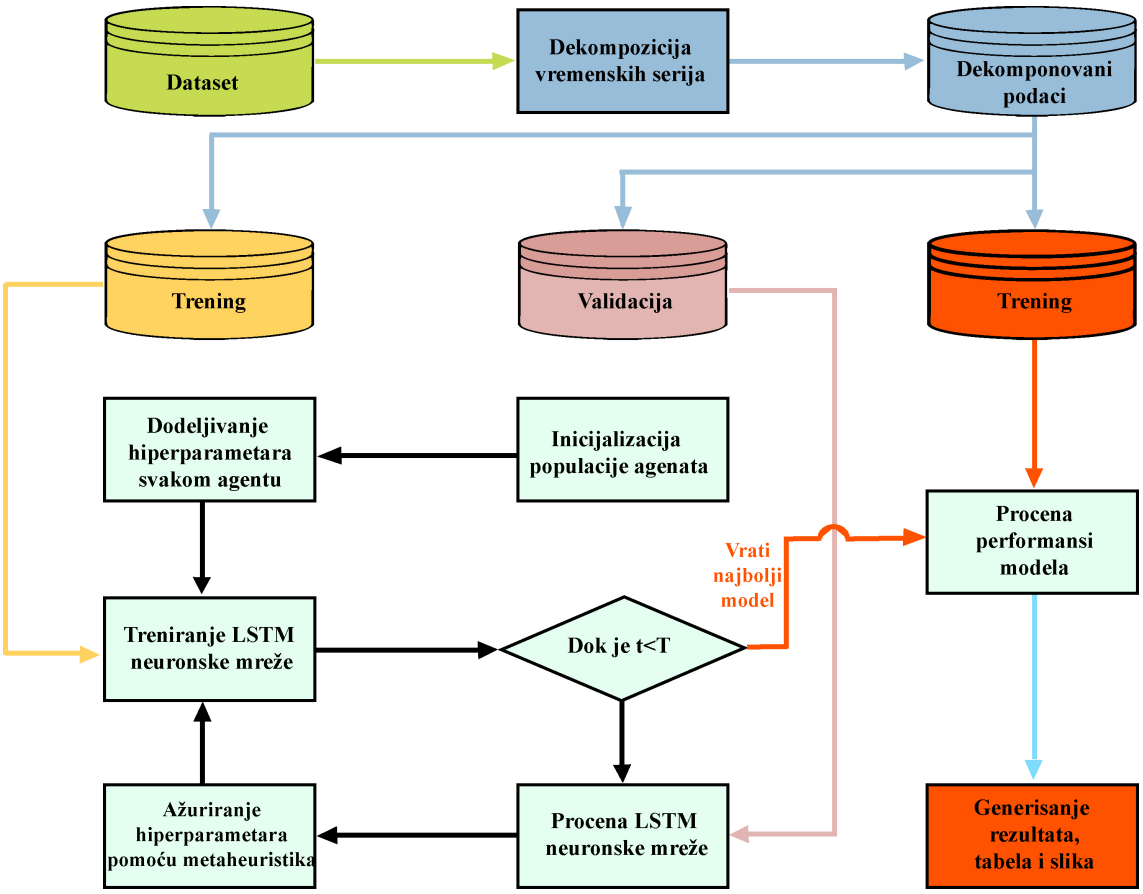
Slika 38: *EMD IMF-ovi i reziduali ulaznih karakteristika za Wind dataset*



Slika 39: *EMD IMF-ovi i reziduali ulaznih karakteristika za Wind farm dataset*

7.9 Postavka eksperimenta

Eksperimentalni proces uključuje dve faze. Prvo, dostupni podaci za oba skupa podataka podvrgnuti su dekompoziciji. Nakon ovog procesa, komponente signala i rezidualni signali dovedeni su u LSTM modele zadužene za predviđanje. Svaki model dobio je šest tačaka ulaznih podataka sa zadatkom da izvrši prognoze tri koraka unapred. Dijagram toka opisanog procesa prikazan je na Slici 40.



Slika 40: *Diagram toka procesa predviđanja*

Nekoliko najsavremenijih metaheuristika korišćeno je za optimizuju parametara modela za predviđanje, kako bi se poboljšale performanse. Evaluirani modeli uključuju predstavljeni ERSA, kao i originalni RSA. Dodatno, u uporednoj analizi uključeni su i neki dobro poznati algoritmi optimizacije, kao što su PSO [70], ABC [68], FA [130], HHO [54], WOA [90], ChOA [71]. Evaluiranim algoritmima optimizacije dodeljen je skup od pet agenata i dozvoljeno je osam iteracija kako bi se poboljšala rešenja. Dodatno, radi uzimanja u obzir prirodne slučajnosti povezane s metaheurističkim algoritmima, rezultati su procenjivani kroz 30 nezavisnih izvršavanja kako bi se postigli objektivni rezultati.

Tabela 4: *Hiperparametri LSTM modela sa odgovarajućim ograničenjima*

naziv parametra	donja granica	gornja granica
stopa učenja	0.0001	0,01
trening epohe	300	600
dropout stopa	0.05	0.2
slojevi mreže	1	2
neuroni u sloju 1	100	300
neuroni u sloju 2	100	300

Svi testirani algoritmi bili su zaduženi za odabir hiperparametara LSTM modela. Podskup parametara odabran za optimizaciju izabran je zbog visokog uticaja na performanse modela. Optimizovani parametri i njihovi mogući opsezi su sledeći: broj neurona u slojevima određen je iz opsega $[100, 300]$, stopa učenja iz opsega $[0.0001, 0.01]$, epohe treniranja iz opsega $[300, 600]$, stopa dropout-a iz opsega $[0.05, 0.2]$, ukupan broj mrežnih slojeva između $[1, 2]$ i broj neurona u drugom sloju mreže između $[100, 300]$. Parametri i njihova odgovarajuća ograničenja prikazani su u Tabeli 4. Konačno, implementirano je ranije zaustavljanje kako bi se sprečilo preprilagođavanje (eng. over-feeting) modela. Prag za ranije zaustavljanje empirijski je određen kao $\frac{epohe}{3}$. To znači da, ako se model ne poboljša u narednih $\frac{epohe}{3}$, treniranje modela se rano prekida. Dodatna prednost ovog pristupa je smanjenje potrošnje računarskih resursa. Korišćeni opsezi odabrani su empirijski kako bi se postigli najbolji rezultati uzimajući u obzir računarske troškove optimizacije i performanse.

Eksperimenti su sprovedeni korišćenjem Python programskog jezika. Dodatno, korišćene su standardne biblioteke za mašinsko učenje i veštačku inteligenciju, uključujući Keras, TensorFlow, Scikit-learn. Za vizualizaciju su korišćene Seaborn i Matplotlib biblioteka. Radi lakšeg izvršavanja eksperimenta, korišćen je računar sa Intel i9 11900K CPU, 128GB RAM memorije i RTX4070 GPU.

8 Postignuti rezultati, uporedna analiza i diskusija

U ovom poglavlju prikazani su eksperimentalni rezultati dobijeni za dva posmatrana skupa podataka: Wind dataset i Wind farm dataset. Prvo su prikazani rezultati bez dekompozicije, a nakon toga rezultati dobijeni primenom VMD i EMD tehnika dekompozicije. Na kraju ovog poglavlja data je i SHAP analiza najboljeg modela za svaki od posmatranih skupova podataka. U svim tabelama koje sadrže eksperimentalne rezultate, najbolji rezultati u svakoj razmatranoj kategoriji označeni su podebljanim (boldiranim) tekstom.

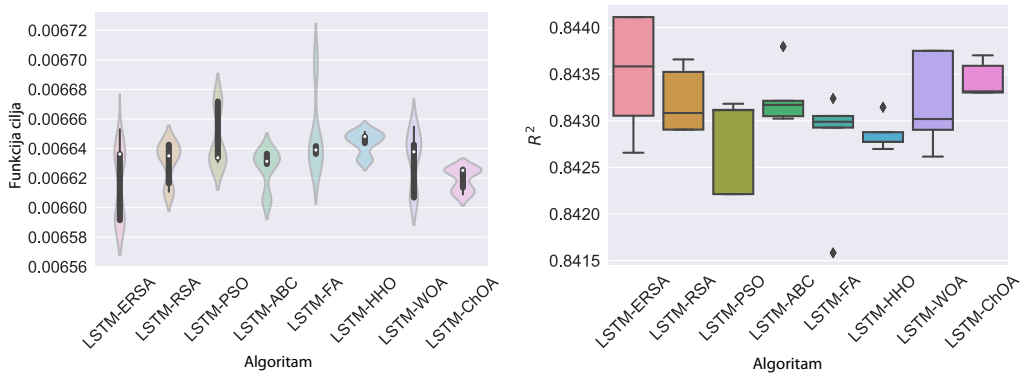
8.1 Eksperimentalni rezultati za Wind dataset

U nastavku su prikazani rezultati za Wind dataset, bez dekompozicije. Tabela 5 prikazuje ukupne metrike u pogledu najboljih, najgorih, srednjih vrednosti i medijana, uz standardnu devijaciju i varijansu tokom 30 različitih izvršavanja svakog posmatranog algoritma. LSTM-ERSA model postigao je najbolje rezultate za najbolje i srednje vrednosti. Drugi najbolji rezultat postignut je pomoću LSTM-ABC metode, dok je LSTM-WOA ostvario treći najbolji rezultat. Sa druge strane, LSTM-ChOA postigao je najbolje rezultate za najgore vrednosti i medijanu. Konačno, LSTM-HHO je ostvario najbolje vrednosti standardne devijacije i varijanse, pri čemu je pokazao najstabilnije rezultate tokom izvršavanja.

Tabela 5: *Wind dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, bez dekompozicije*

Metod	Najbolji	Najgori	Srednji	Medijana	Std	Var
LSTM-ERSA	**0.006592**	0.006653	**0.006616**	0.006636	2.30E-05	5.30E-10
LSTM-RSA	0.006611	0.006643	0.006631	0.006635	1.27E-05	1.61E-10
LSTM-PSO	0.006631	0.006672	0.006646	0.006634	1.83E-05	3.36E-10
LSTM-ABC	0.006605	0.006638	0.006627	0.006631	1.24E-05	1.54E-10
LSTM-FA	0.006628	0.006698	0.006649	0.006639	2.53E-05	6.38E-10
LSTM-HHO	0.006632	0.006651	0.006644	0.006648	**7.00E-06**	**4.90E-11**
LSTM-WOA	0.006607	0.006655	0.006633	0.006638	1.82E-05	3.30E-10
LSTM-ChOA	0.006609	**0.006626**	0.006619	**0.006625**	7.32E-06	5.36E-11

U Tabeli 6 prikazane su detaljne metrike za svaki korak predviđanja sa fokusom na najbolje izvršavanje svakog algoritma. Prefiks L se koristi kako bi označio korišćenje LSTM-a. Primećuje se da je predloženi LSTM-ERSA postigao najbolje rezultate za dva i tri koraka unapred, i najbolje ukupne rezultate, u pogledu funkcije cilja - MSE, ali i za druge važne pokazatelje, kao što su R^2, MAE (osim u slučaju za dva koraka



Slika 41: *Wind dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku bez dekompozicije*

unapred) i RMSE. Najbolje rezultate za jedan korak unapred postigao je LSTM-ABC algoritam. U pogledu ukupnih rezultata, drugi najbolji algoritam bio je LSTM-ABC, ispred metoda LSTM-WOA i LSTM-ChOA. Posmatrani LSTM-ERSA postigao je najbolju ukupnu vrednost MSE od 0.006591, ispred LSTM-ABC sa vrednošću MSE od 0.006604.

Najbolji skup LSTM parametara dobijen za najbolje izvršavanje, svake metaheuristike prikazan je u Tabeli 7. LSTM parametri dobijeni pomoću predloženog LSTM-ERSA modela su sledeći: 140 neurona u prvom sloju, stopa učenja od 0.000894, 517 epoha, vrednost dropout-a od 0.108587, i 170 neurona u drugom sloju.

Kako bi se dobio bolji vizuelni uvid u rezultate date su Slike od 41 do 43. Slika 41 prikazuje violin dijagrame za funkciju cilja (MSE), uz prateći box dijagram za indikator R^2, za svih 30 izvršavanja. Dijagrami konvergencije funkcije cilja i R^2 za najbolje izvršavanje svakog algoritma prikazani su na Slici 42. Primećuje se da su na početku metaheuristike RSA i WOA brže konvergirale, ali je u poslednjim rundama izvršavanja ERSA postigao bolje rezultate. Na kraju, Kernel dijagram (eng. Kernel Density Estimation - KDE) i dijagram roja (eng. swarm plot) prikazani su na Slici 43. KDE dijagram se koristi za prikaz funkcije gustine verovatnoće i može pokazati da li rezultati dolaze iz normalne raspodele. Swarm dijagram prikazuje raznovrsnost rešenja tokom poslednje runde najboljeg izvršavanja svakog algoritma.

8.1.1 Wind dataset sa VMD dekompozicijom

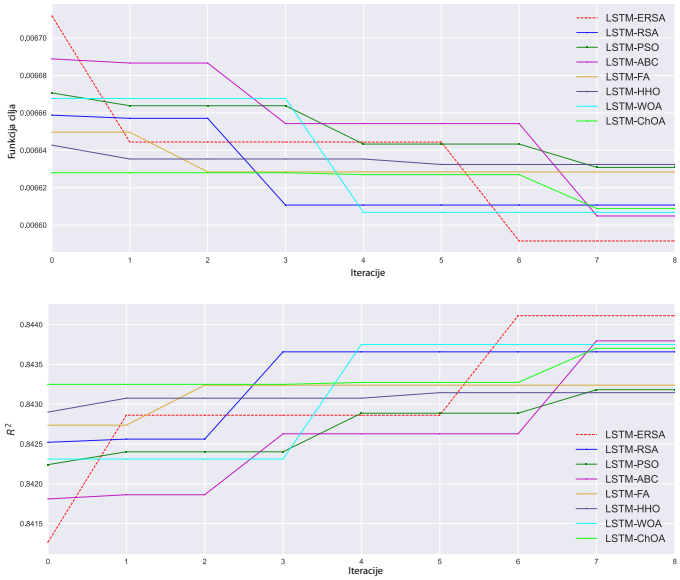
Ovaj deo predstavlja eksperimentalne rezultate za Wind dataset kada je primenjena VMD dekompozicija. Tabela 8 prikazuje ukupne metrike u pogledu najboljih,

Tabela 6: *Wind dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje bez dekompozicije*

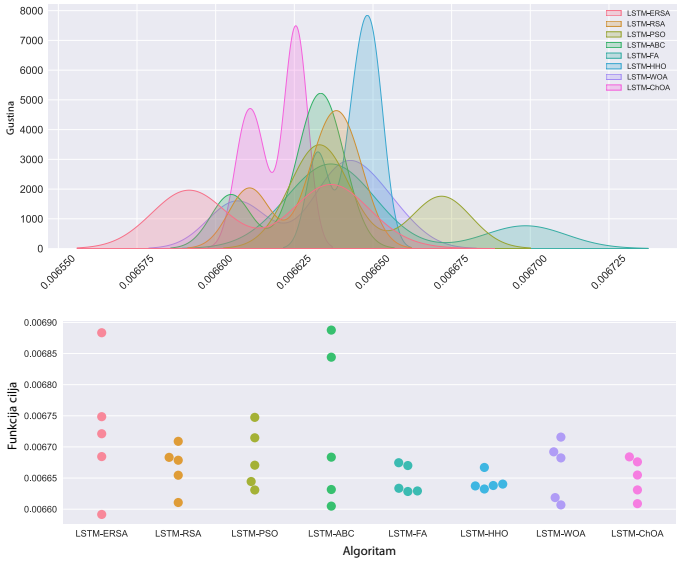
	Indikator greške	L-ERSA	L-RSA	L-PSO	L-ABC	L-FA	L-HHO	L-WOA	L-ChOA
Jedan	R^2	**0.845391**	0.845365	0.844410	0.845795	0.843997	0.844825	0.845621	0.845740
korak	MAE	0.051652	0.051778	0.051980	0.051555	0.051913	0.051905	0.051622	**0.051469**
unapred	MSE	0.006537	0.006539	0.006579	**0.006520**	0.006596	0.006561	0.006528	0.006523
	RMSE	0.080855	0.080861	0.081110	**0.080749**	0.081218	0.081002	0.080794	0.080763
Dva	R^2	**0.845174**	0.844677	0.844587	0.844842	0.844313	0.844381	0.844986	0.844649
koraka	MAE	0.052106	0.052294	0.052345	0.052263	0.052233	0.052475	**0.052025**	0.052158
unapred	MSE	**0.006547**	0.006568	0.006571	0.006561	0.006583	0.006580	0.006555	0.006569
	RMSE	**0.080911**	0.081041	0.081064	0.080998	0.081136	0.081118	0.080960	0.081048
Tri	R^2	**0.841770**	0.840932	0.840548	0.840750	0.841407	0.840229	0.840642	0.840716
koraka	MAE	**0.053340**	0.053716	0.053675	0.053775	0.053446	0.053942	0.053581	0.053602
unapred	MSE	**0.006690**	0.006725	0.006741	0.006733	0.006705	0.006755	0.006737	0.006734
	RMSE	**0.081795**	0.082011	0.082110	0.082058	0.081888	0.082192	0.082085	0.082067
Ukupni	R^2	**0.844111**	0.843657	0.843181	0.843795	0.843238	0.843144	0.843749	0.843701
rezultati	MAE	**0.052365**	0.052596	0.052666	0.052530	0.052530	0.052773	0.052409	0.052409
predviđanja	MSE	**0.006591**	0.006610	0.006630	0.006604	0.006627	0.006631	0.006606	0.006608
	RMSE	**0.081187**	0.081305	0.081429	0.081269	0.081414	0.081439	0.081281	0.081294

Tabela 7: *Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra bez dekompozicije*

Metod	Neuroni u sloju 1	Stopa učenja	Epohe	Dropout	Slojevi	Neuroni u sloju 2
LSTM-ERSA	140	0.000894	517	0.108587	2	170
LSTM-RSA	100	0.003232	318	0.091040	2	150
LSTM-PSO	100	0.002880	300	0.050317	2	200
LSTM-ABC	167	0.001091	547	0.081649	2	188
LSTM-FA	100	0.010000	300	0.200000	1	118
LSTM-HHO	184	0.002868	346	0.175802	2	158
LSTM-WOA	131	0.001016	600	0.055888	2	129
LSTM-ChOA	100	0.003440	592	0.183099	1	177



Slika 42: *Wind dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku bez dekompozicije*



Slika 43: *Wind dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku bez dekompozicije*

Tabela 8: *Wind dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, sa primenom VMD dekompozicije*

Metod	Najbolji	Najgori	Srednj	Medijana	Std	Var
VMD-LSTM-ERSA	**0.001704**	0.001841	**0.001750**	**0.001733**	5.16E-05	2.66E-09
VMD-LSTM-RSA	0.001729	0.001817	0.001784	0.001800	3.23E-05	1.05E-09
VMD-LSTM-PSO	0.001759	0.001879	0.001813	0.001809	4.39E-05	1.93E-09
VMD-LSTM-ABC	0.001779	0.001866	0.001809	0.001792	3.79E-05	1.44E-09
VMD-LSTM-FA	0.001730	0.001814	0.001756	0.001743	3.21E-05	1.03E-09
VMD-LSTM-HHO	0.001758	**0.001811**	0.001771	0.001761	**2.19E-05**	**4.80E-10**
VMD-LSTM-WOA	0.001762	0.001820	0.001784	0.001765	2.69E-05	7.25E-10
VMD-LSTM-ChOA	0.001726	0.001824	0.001766	0.001743	3.68E-05	1.35E-09

najgorih, srednjih vrednosti i medijana, uz standardnu devijaciju i varijansu tokom 30 različitih izvršavanja svakog posmatranog algoritma. VMD-LSTM-ERSA model postigao je najbolje rezultate za najbolje, srednje vrednosi i medijane. Drugi najbolji rezultat postignut je pomoću metode VMD-LSTM-ChOA, dok je VMD-LSTM-RSA ostvario treći najbolji rezultat. Sa druge strane, VMD-LSTM-HHO postigao je najbolji rezultat za najgoru vrednost. Konačno, VMD-LSTM-HHO je takođe postavio najbolje vrednosti standardne devijacije i varijanse, pri čemu je pokazao najstabilnije rezultate tokom 30 nezavisnih izvršavanja.

U Tabeli 9 prikazane su detaljne metrike za svaki korak predviđanja sa fokusom na najbolje izvršavanje svakog algoritma. Prefiks VL se koristi kako bi označilo korišćenje VMD-LSTM-a. Primećuje se da je predloženi VMD-LSTM-ERSA postigao izuzetne rezultate za jedan korak unapred i ukupne rezultate, u pogledu funkcije cilja - MSE, ali i za druge važne pokazatelje, kao što su R^2, MAE i RMSE. Najbolje rezultate za dva koraka unapred, postigao je VMD-LSTM-WOA, dok je VMD-LSTM-HHO postigao najbolje rezultate za tri koraka unapred. Gledajući ukupne rezultate, drugi najbolji algoritam bio je VMD-LSTM-ChOA, ispred metoda VMD-LSTM-RSA. Posmatrani VMD-LSTM-ERSA postigao je najbolju ukupnu vrednost MSE od 0.001703, ispred VMD-LSTM-ChOA sa vrednošću MSE od 0.001725.

Najbolji skup LSTM parametara dobijen za najbolje izvršavanje, svake metaheuristike prikazan je u Tabeli 10. LSTM parametri dobijeni pomoću predloženog VMD-LSTM-ERSA modela su sledeći: 100 neurona u prvom sloju, stopa učenja od 0.010000, 563 epoha, vrednost dropout-a od 0.200000 i 100 neurona u drugom sloju.

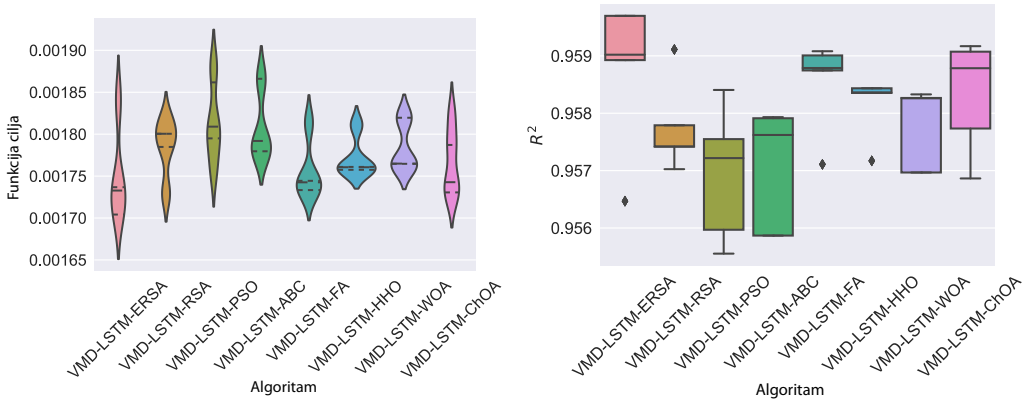
Kako bi se dobio bolji vizuelni uvid u rezultate date su Slike od od 44 do 46. Slika 44 prikazuje violin dijagrame za funkciju cilja (MSE), uz prateći box dijagram za indikator R^2, za svih 30 izvršavanja. Dijagrami konvergencije funkcije cilja i R^2

Tabela 9: *Wind dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje sa primenom VMD dekompozicije*

	Indikator geške	VL-ERSA	VL-RSA	VL-PSO	VL-ABC	VL-FA	VL-HHO	VL-WOA	VL-ChOA
Jedan	R^2	**0.941916**	0.940073	0.937082	0.938810	0.939588	0.936426	0.935198	0.938841
korak	MAE	**0.032073**	0.032604	0.033309	0.032766	0.032744	0.033708	0.033620	0.032859
unapred	MSE	**0.002456**	0.002534	0.002660	0.002587	0.002554	0.002688	0.002740	0.002586
	RMSE	**0.049558**	0.050338	0.051579	0.050866	0.050541	0.051847	0.052346	0.050853
Dva	R^2	0.963863	0.965451	0.965805	0.962887	0.964136	0.964482	**0.965907**	0.964385
koraka	MAE	0.023949	0.023585	0.023410	0.024447	0.024082	0.023929	**0.023259**	0.023829
unapred	MSE	0.001528	0.001461	0.001446	0.001569	0.001516	0.001502	**0.001442**	0.001506
	RMSE	0.039090	0.038221	0.038025	0.039614	0.038942	0.038753	**0.037968**	0.038806
Tri	R^2	0.973313	0.971810	0.972330	0.972100	0.973517	**0.974396**	0.973880	0.974280
koraka	MAE	0.020805	0.021806	0.021357	0.021592	0.020839	**0.020161**	0.020653	0.020345
ubapred	MSE	0.001127	0.001191	0.001169	0.001179	0.001119	**0.001082**	0.001103	0.001086
	RMSE	0.033590	0.034523	0.034203	0.034346	0.033462	**0.032902**	0.033232	0.032976
Ukupni	R^2	**0.959696**	0.959111	0.958405	0.957931	0.959080	0.958434	0.958328	0.959168
rezultati	MAE	**0.025608**	0.025998	0.026025	0.026268	0.025888	0.025932	0.025843	0.025677
predviđanja	MSE	**0.001703**	0.001728	0.001758	0.001778	0.001729	0.001757	0.001761	0.001725
	RMSE	**0.041280**	0.041579	0.041936	0.042175	0.041595	0.041922	0.041975	0.041550

Tabela 10: *Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra sa primenom VMD dekompozicije*

Metod	Neuroni u sloju 1	Stopa učenja	Epohe	Dropout	Slojevi	Neuroni u sloju 2
VMD-LSTM-ERSA	100	0.010000	563	0.200000	2	100
VMD-LSTM-RSA	100	0.010000	600	0.073598	2	104
VMD-LSTM-PSO	139	0.007558	600	0.050000	2	138
VMD-LSTM-ABC	174	0.006837	509	0.164527	2	139
VMD-LSTM-FA	143	0.010000	600	0.098000	2	130
VMD-LSTM-HHO	171	0.008113	572	0.200000	2	162
VMD-LSTM-WOA	200	0.005682	600	0.055617	2	200
VMD-LSTM-ChOA	100	0.008296	600	0.050000	2	146

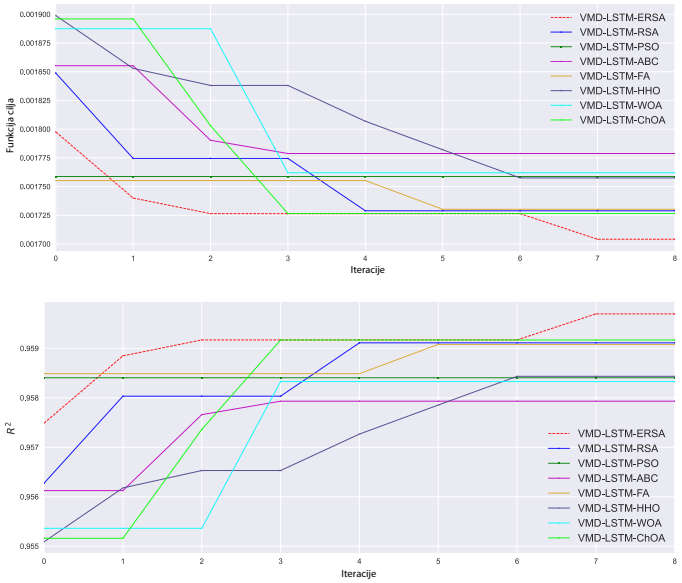


Slika 44: *Wind dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku sa primenom VMD dekompozicije*

za najbolje izvršavanje svakog algoritma prikazani su na Slici 45. Može se primetiti se da u ovom slučaju predloženi ERSA, od početka do kraja, pokazuje najbržu konvergenciju. KDE i swarm dijagrami za funkciju cilja prikazani su na Slici 46.

8.1.2 Wind dataset sa EMD dekompozicijom

Ovaj deo predstavlja eksperimentalne rezultate za Wind dataset kada je primenjena EMD tehnika dekompozicije. Tabela 11 prikazuje ukupne metrike u pogledu najboljih, najgorih, srednjih vrednosti i medijana, uz standardnu devijaciju i varijansu tokom 30 različitih izvršavanja svakog posmatranog algoritma. EMD-LSTM-ERSA model postigao je izuzetne rezultate za najbolje, srednje vrednosi i medijane. Drugi najbolji rezultat postignut je pomoću EMD-LSTM-PSO metode, dok je EMD-LSTM-FA ostvario treći najbolji rezultat. Sa druge strane, EMD-LSTM-PSO postigao je najbolji rezultat za najgoru metriku. Konačno, EMD-LSTM-RSA je postavio najbolje vrednosti standardne devijacije i varijanse, pri čemu je pokazao najstabilnije rezultate tokom 30 nezavisnih izvršavanja za ovaj slučaj.



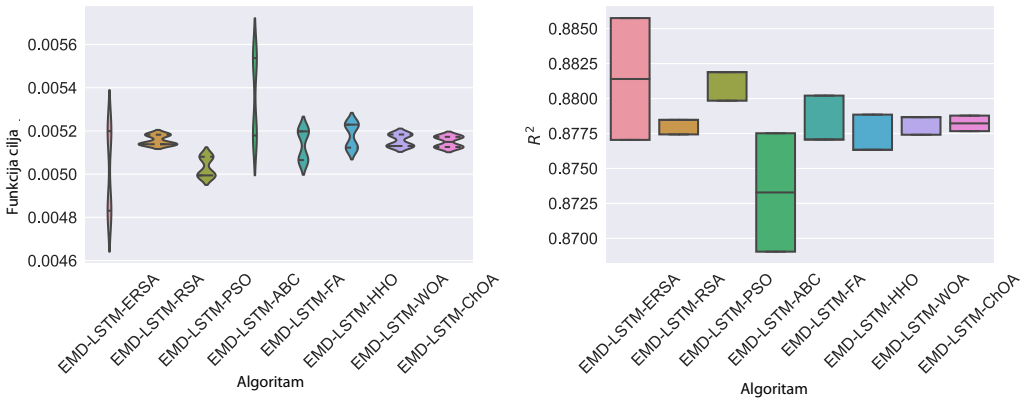
Slika 45: *Wind dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku sa primenom VMD dekompozicije*



Slika 46: *Wind dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku sa primenom VMD dekompozicije*

Tabela 11: *Wind dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, sa primenom EMD dekompozicije*

Metod	Najbolji	Najgori	Srednji	Medijana	Std	Var
EMD-LSTM-ERSA	**0.004831**	0.005199	**0.005015**	**0.005015**	1.84E-04	3.39E-08
EMD-LSTM-RSA	0.005139	0.005183	0.005156	0.005139	**2.15E-05**	**4.64E-10**
EMD-LSTM-PSO	0.004994	**0.005081**	0.005032	0.004994	4.28E-05	1.83E-09
EMD-LSTM-ABC	0.005179	0.005537	0.005358	0.005358	1.79E-04	3.20E-08
EMD-LSTM-FA	0.005065	0.005198	0.005136	0.005198	6.62E-05	4.39E-09
EMD-LSTM-HHO	0.005123	0.005229	0.005179	0.005229	5.31E-05	2.82E-09
EMD-LSTM-WOA	0.005130	0.005184	0.005155	0.005130	2.66E-05	7.06E-10
EMD-LSTM-ChOA	0.005126	0.005173	0.005149	0.005149	2.35E-05	5.52E-10



Slika 47: *Wind dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku sa primenom EMD dekompozicije*

U Tabeli 12 prikazane su detaljne metrike za svaki korak predviđanja sa fokusom na najbolje izvršavanje svakog algoritma. Prefiks EL se koristi kako bi označilo korišćenje EMD-LSTM-a. U ovom slučaju, primećuje se da je predloženi EMD-LSTM-ERSA postigao izuzetne rezultate za sve metrike: jedan, dva i tri koraka unapred, kao i za ukupni rezultat, u pogledu funkcije cilja - MSE, ali i svih ostalih važnih pokazatelja, kao što su R^2, MAE i RMSE. Ukoliko se posmatraju ukupni rezultati, drugi najbolji algoritam bio je EMD-LSTM-PSO, ispred metoda EMD-LSTM-FA. Predloženi EMD-LSTM-ERSA postigao je najbolju ukupnu vrednost MSE od 0.004830, ispred EMD-LSTM-PSO koji je postigao vrednost MSE od 0.004993.

Najbolji skup LSTM parametara dobijen za najbolje izvršavanje, svake metaheuristike prikazan je u slučaju kada je primenjena EMD dekompozicija, prikazan je u Tabeli 13. LSTM parametri dobijeni pomoću predloženog EMD-LSTM-ERSA modela su sledeći: 108 neurona u prvom sloju, stopa učenja od 0.007837, 600 epoha, vrednost dropout-a od 0.050000, i 151 neurona u drugom sloju.

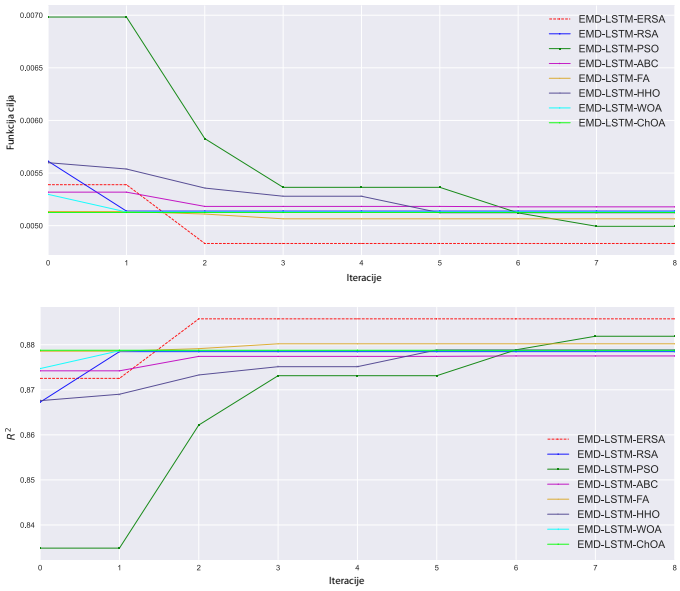
S ciljem pružanja boljeg vizuelnog uvida u rezultate, prikazani su dijagrami na Slikama od 47 do 49. Slika 47 prikazuje violin dijagrame za funkciju cilja (MSE), uz prateći box dijagram za indikator R^2, za svih 30 izvršavanja. Dijagrami konvergencije funkcije cilja i R^2 za najbolje izvršavanje svakog algoritma prikazani su na Slici 48. Može se primetiti da u ovom slučaju predloženi ERSA pokazuje najbržu konvergenciju od početka do kraja. Na kraju, KDE i swarm dijagramima za funkciju cilja, prikazani su na Slici 49.

Tabela 12: *Wind dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje sa primenom EMD dekompozicije*

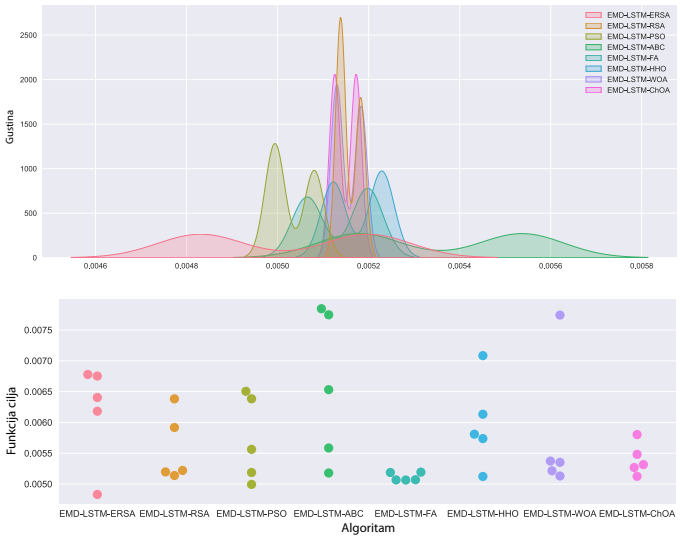
	Indikator geške	EL-ERSA	EL-RSA	EL-PSO	EL-ABC	EL-FA	EL-HHO	EL-WOA	EL-ChOA
Jedan	R^2	**0.882618**	0.874563	0.876980	0.877066	0.875932	0.870262	0.872794	0.875291
korak	MAE	**0.046090**	0.048054	0.047590	0.047374	0.047323	0.048471	0.048793	0.048225
unapred	MSE	**0.004963**	0.005304	0.005202	0.005198	0.005246	0.005486	0.005379	0.005273
	RMSE	**0.070451**	0.072828	0.072123	0.072098	0.072430	0.074066	0.073340	0.072617
Dva	R^2	**0.886238**	0.876369	0.883291	0.871967	0.882816	0.884023	0.882208	0.879532
koraka	MAE	**0.045484**	0.047946	0.046415	0.048618	0.045821	0.045980	0.046948	0.047558
unapred	MSE	**0.004810**	0.005228	0.004935	0.005414	0.004955	0.004904	0.004981	0.005094
	RMSE	**0.069356**	0.072302	0.070249	0.073578	0.070392	0.070028	0.070574	0.071371
Tri	R^2	**0.888399**	0.884489	0.885392	0.883520	0.881879	0.882265	0.881007	0.881513
koraka	MAE	**0.045805**	0.046869	0.046693	0.046724	0.046394	0.047158	0.047604	0.047710
unapred	MSE	**0.004718**	0.004883	0.004845	0.004924	0.004994	0.004977	0.005030	0.005009
	RMSE	**0.068693**	0.069886	0.069612	0.070189	0.070671	0.070556	0.070931	0.070780
Ukupni	R^2	**0.885751**	0.878473	0.881887	0.877517	0.880208	0.878849	0.878669	0.878778
rezultati	MAE	**0.045792**	0.047622	0.046899	0.047571	0.046512	0.047202	0.047781	0.047830
predviđanja	MSE	**0.004830**	0.005138	0.004993	0.005178	0.005064	0.005122	0.005129	0.005125
	RMSE	**0.069503**	0.071683	0.070669	0.071964	0.071169	0.071572	0.071625	0.071593

Tabela 13: *Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra sa primenom EMD dekompozicije*

Metod	Neuroni u sloju 1	Stopa učenja	Epohe	Dropout	Slojevi	Neuroni u sloju 2
EMD-LSTM-ERSA	108	0.007837	600	0.050000	2	151
EMD-LSTM-RSA	200	0.010000	600	0.119344	2	100
EMD-LSTM-PSO	121	0.008541	600	0.173318	2	164
EMD-LSTM-ABC	165	0.007747	600	0.119184	2	133
EMD-LSTM-FA	155	0.010000	600	0.200000	2	101
EMD-LSTM-HHO	106	0.006599	570	0.062141	2	112
EMD-LSTM-WOA	100	0.010000	600	0.184404	2	118
EMD-LSTM-ChOA	136	0.004410	549	0.114572	2	104



Slika 48: *Wind dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku sa primenom EMD dekompozicije*



Slika 49: *Wind dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku sa primenom EMD dekompozicije*

Tabela 14: *Poređenje najboljeg modela sa savremenim modelima za predikciju koji su primenjeni na Wind dataset*

Metod	MSE	R^2
SVM	0.007283	0.814233
Random forest	0.007303	0.772332
XGBoost	0.007126	0.807384
KELM	0.007200	0.801634
AdaBoost	0.007203	0.802330
ANN (3 sloja)	0.007190	0.791192
RNN (3 sloja)	0.007199	0.805466
LSTM	0.007233	0.824233
LSTM-ERSA	0.006591	0.844111
VMD-LSTM-ERSA	**0.001703**	**0.959696**
EMD-LSTM-ERSA	0.004830	0.885751

8.1.3 Poređenje sa drugim modelima za Wind dataset

Da bi se demonstrirala komparativna poboljšanja performansi optimizatora spregnutih sa tehnikama dekompozicije u uvedenoj metodologiji, najbolji rezultati svakog pristupa su upoređeni sa nekoliko savremenih modela predviđanja. Rezultati funkcije cilja (MSE) i indikatora R^2 prikazani su u Tabeli 14.

Kao što je prikazano u Tabeli 14, rezultati postignuti primenom VMD tehnike, praćeni LSTM mrežama optimizovanim pomoću uvedenih metaheuristika, pokazuju značajna poboljšanja u poređenju sa drugim pristupima koji su primenjeni na istom zadatku.

8.2 Eksperimentalni rezultati za Wind farm dataset

Ovaj deo predstavlja eksperimentalne rezultate za Wind farm dataset, bez dekompozicije. Tabela 15 prikazuje ukupne metrike u pogledu najboljih, najgorih, srednjih vrednosti i medijana, uz standardnu devijaciju i varijansu tokom 30 različitih izvršavanja svakog posmatranog algoritma. LSTM-ERSA model postigao je izuzetne rezultate za najbolje, najgore i srednje vrednosti. Drugi najbolji rezultat postignut je pomoću LSTM-HHO metode, dok je LSTM-ABC ostvario treći najbolji rezultat. Sa druge strane, LSTM-ABC postigao je najbolje rezultate za medijanu. Konačno, LSTM-RSA postavio je najbolje vrednosti standardne devijacije i varijanse, čime je pokazao najstabilnije rezultate tokom 30 nezavisnih izvršavanja.

U Tabeli 16 prikazane su detaljne metrike za svaki korak predviđanja sa fokusom na najbolje izvršavanje svakog algoritma. Prefiks L se koristi kako bi označio korišćenje

LSTM-a. Primećuje se da je predloženi LSTM-ERSA postigao izuzetne rezultate za dva koraka unapred i za ukupne rezultate, u pogledu funkcije cilja - MSE, ali i za druge važne pokazatelje, kao što su R^2 i RMSE (osim MAE). Najbolji rezultati za jedan korak unapred postignuti su pomoću LSTM-HHO pristupa, dok su najbolji rezultati za tri koraka unapred postignuti pomoću LSTM-ChOA. Posmatranjem ukupnih rezultata, drugi najbolji algoritam bio je LSTM-HHO, ispred metoda LSTM-ABC i LSTM-ChOA. Posmatrani LSTM-ERSA postigao je najbolju ukupnu vrednost MSE od 0.020565, ispred LSTM-HHO sa vrednošću MSE od 0.020607.

Tabela 15: *Wind farm dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, bez dekompozicije*

Metod	Najbolji	Najgori	Srednji	Medijana	Std	Var
LSTM-ERSA	**0.020566**	**0.020809**	**0.020724**	0.020803	9.69E-05	9.38E-09
LSTM-RSA	0.020829	0.020884	0.020864	0.020872	**1.88E-05**	**3.54E-10**
LSTM-PSO	0.020754	0.020847	0.020809	0.020821	3.43E-05	1.18E-09
LSTM-ABC	0.020673	0.020857	0.020745	**0.020722**	6.59E-05	4.34E-09
LSTM-FA	0.020747	0.021040	0.020911	0.020983	1.10E-04	1.22E-08
LSTM-HHO	0.020608	0.020925	0.020807	0.020869	1.23E-04	1.52E-08
LSTM-WOA	0.020730	0.020881	0.020807	0.020826	5.68E-05	3.23E-09
LSTM-ChOA	0.020699	0.020947	0.020775	0.020727	8.98E-05	8.07E-09

Najbolji skup LSTM parametara dobijen za najbolje izvršavanje, svake metaheuristike prikazan je u Tabeli 17. LSTM parametri dobijeni pomoću predloženog LSTM-ERSA modela su sledeći: 180 neurona u prvom sloju, stopa učenja od 0.005919, 439 epoha, vrednost dropout-a od 0.165101, i 200 neurona u drugom sloju.

S ciljem pružanja boljeg vizuelnog uvida u rezultate, prikazani su dijagrami na Slikama od 50 do 52. Slika 50 prikazuje violin dijagrame za funkciju cilja (MSE), uz prateći box dijagram za indikator R^2, za svih 30 izvršavanja. Dijagrami konvergencije funkcije cilja i R^2 za najbolje izvršavanje svakog algoritma prikazani su na Slici 51. Primećuje se da je na početku metaheuristika HHO brže konvergirala, ali je u poslednjim rundama izvršavanja ERSA postigao bolje rezultate. Na kraju su, KDE dijagram i swarm dijagrami prikazani na Slici 52.

8.2.1 Wind farm dataset sa VMD dekompozicijom

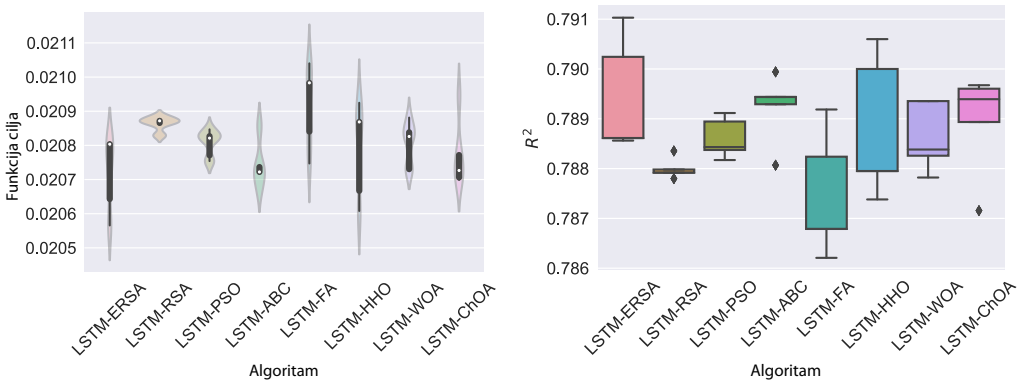
U ovom delu predstavljeni su eksperimentalne rezultate za Wind farm dataset kada je primenjena VMD dekompozicija. Tabela 18 prikazuje ukupne metrike u pogledu najboljih, najgorih, srednjih vrednosti i medijana, uz standardnu devijaciju i

Tabela 16: *Wind farm dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje bez dekompozicije*

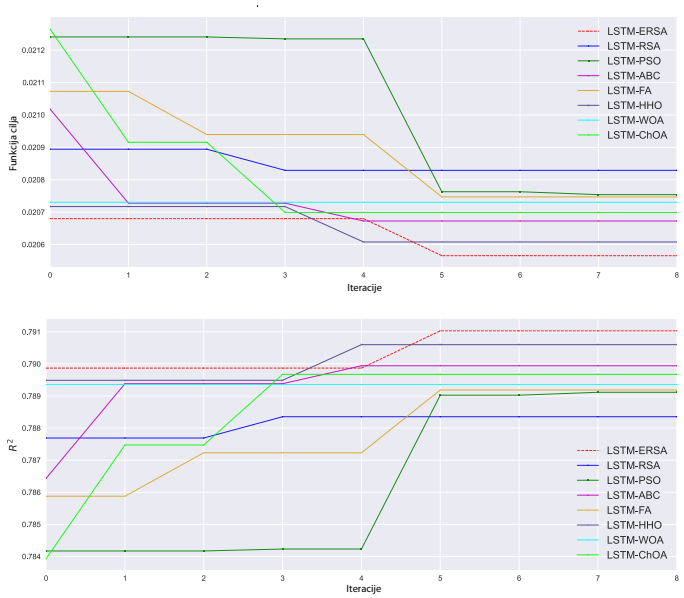
	Indikator geške	L-ERSA	L-RSA	L-PSO	L-ABC	L-FA	L-HHO	L-WOA	L-ChOA
Jedan	R^2	0.771359	0.768250	0.770386	0.769249	0.769645	**0.775001**	0.769093	0.767537
korak	MAE	**0.102667**	0.102964	0.102777	0.102829	0.102680	0.102882	0.102924	0.102791
unapred	MSE	0.022502	0.022808	0.022597	0.022709	0.022670	**0.022143**	0.022724	0.022878
	RMSE	0.150005	0.151022	0.150324	0.150696	0.150566	**0.148806**	0.150746	0.151254
Dva	R^2	**0.799068**	0.796724	0.796229	0.797961	0.796895	0.796674	0.797062	0.796279
koraka	MAE	0.100233	0.099696	0.100523	0.099993	0.100032	0.101472	0.100125	**0.099405**
unapred	MSE	**0.019775**	0.020005	0.020054	0.019884	0.019988	0.020010	0.019972	0.020049
	RMSE	**0.140622**	0.141440	0.141612	0.141009	0.141381	0.141457	0.141322	0.141595
Tri	R^2	0.802662	0.800087	0.800731	0.802615	0.801019	0.800123	0.801910	**0.805205**
koraka	MAE	0.103533	0.103045	0.103602	0.102485	0.102978	0.104569	0.103026	**0.100661**
unapred	MSE	0.019420	0.019673	0.019610	0.019424	0.019581	0.019670	0.019494	**0.019170**
	RMSE	0.139357	0.140263	0.140038	0.139374	0.139936	0.140251	0.139623	**0.138457**
Ukupni	R^2	**0.791029**	0.788353	0.789115	0.789941	0.789186	0.790598	0.789354	0.789673
rezultati	MAE	0.102144	0.101901	0.102300	0.101768	0.101896	0.102974	0.102024	**0.100951**
predviđanja	MSE	**0.020565**	0.020828	0.020753	0.020672	0.020746	0.020607	0.020729	0.020698
	RMSE	**0.143406**	0.144321	0.144061	0.143779	0.144037	0.143554	0.143980	0.143871

Tabela 17: *Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra bez dekompozicije*

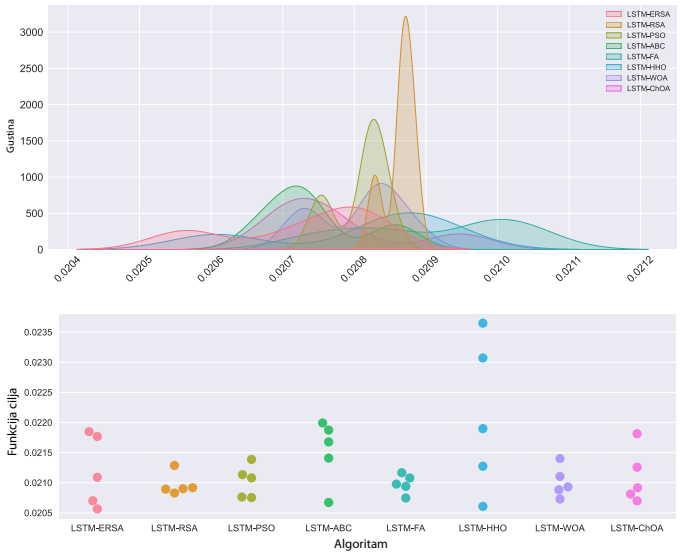
Metod	Neuroni u sloju 1	Stopa učenja	Epohe	Dropout	Slojevi	Neuroni u sloju 2
LSTM-ERSA	180	0.005919	439	0.165101	2	200
LSTM-RSA	137	0.010000	553	0.200000	2	156
LSTM-PSO	200	0.005864	480	0.196612	2	176
LSTM-ABC	117	0.007639	527	0.099844	2	142
LSTM-FA	186	0.010000	578	0.200000	2	151
LSTM-HHO	176	0.006243	485	0.108088	2	150
LSTM-WOA	197	0.006295	543	0.198430	2	186
LSTM-ChOA	100	0.010000	473	0.050467	2	134



Slika 50: *Wind farm dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku bez dekompozicije*



Slika 51: *Wind farm dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku bez dekompozicije*



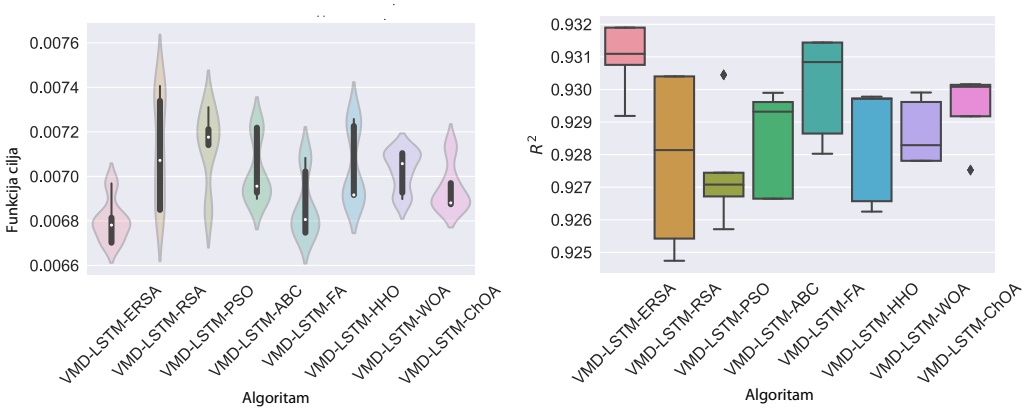
Slika 52: *Wind farm dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku bez dekompozicije*

Tabela 18: *Wind farm dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, sa primenom VMD dekompozicije*

Metod	Najbolji	Najgori	Srednji	Medijana	Std	Var
VMD-LSTM-ERSA	**0.006702**	**0.006969**	**0.006802**	**0.006781**	**8.53E-05**	7.96E-09
VMD-LSTM-RSA	0.006849	0.007406	0.007135	0.007072	2.24E-04	5.00E-08
VMD-LSTM-PSO	0.006845	0.007311	0.007140	0.007176	1.60E-04	2.56E-08
VMD-LSTM-ABC	0.006899	0.007219	0.007021	0.006956	1.33E-04	1.76E-08
VMD-LSTM-FA	0.006747	0.007083	0.006867	0.006806	1.35E-04	1.81E-08
VMD-LSTM-HHO	0.006911	0.007258	0.007046	0.006917	1.61E-04	2.59E-08
VMD-LSTM-WOA	0.006898	0.007104	0.007027	0.007057	8.60E-05	**7.40E-09**
VMD-LSTM-ChOA	0.006873	0.007132	0.006947	0.006881	9.95E-05	9.90E-09

varijansu tokom 30 različitih izvršavanja svakog posmatranog algoritma. VMD-LSTM-ERSA model postigao je izuzetne rezultate za najbolje, najgore, srednje vrednosi i medijane. Drugi najbolji rezultat postignut je pomoću VMD-LSTM-FA metoda, dok je VMD-LSTM-PSO ostvario treći najbolji rezultat. VMD-LSTM-ERSA postigao je i najbolje rezultate za standardnu devijaciju, dok je LSTM-WOA postigao najbolju vrednost varijanse.

U Tabeli 19 prikazane su detaljne metrike za svaki korak predviđanja sa fokusom na najbolje izvršavanje svakog algoritma. Prefiks VL se koristi kako bi označio korišćenje VMD-LSMT-a. Primećuje se da je predloženi VMD-LSTM-ERSA postigao najbolje vrednosti za ukupne rezultate, u pogledu funkcije cilja - MSE, ali i za druge važne pokazatelje, kao što su R^2 i RMSE (osim MAE). Najbolji rezultati za jedan korak unapred postignuti su pomoću VMD-LSTM-HHO pristupa, najbolji rezultati za dva koraka unapred dobijeni su pomoću VMD-LSTM-RSA, dok su



Slika 53: *Wind farm dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku sa primenom VMD dekompozicije*

najbolji rezultati za tri koraka unapred ostvareni pomoću VMD-LSTM-FA. U pogledu ukupnih rezultata, drugi najbolji algoritam bio je VMD-LSTM-FA, ispred VMD-LSTM-PSO. Predloženi VMD-LSTM-ERSA postigao je najbolju ukupnu vrednost MSE od 0.006701, ispred VMD-LSTM-FA sa vrednošću MSE od 0.006746.

Najbolji skup LSTM parametara dobijen za najbolje izvršavanje, svake metaheuristike prikazan je u Tabeli 20. LSTM parametri dobijeni pomoću predloženog VMD-LSTM-ERSA postavio modela su sledeći: 142 neurona u prvom sloju, stopa učenja od 0.009502, 600 epoha, vrednost dropout-a od 0.151008, i 127 neurona u drugom sloju.

S ciljem pružanja boljeg vizuelnog uvida u rezultate, prikazani su dijagrami na Slikama od 53 - 55. Slika 53 prikazuje violin dijagrame za funkciju cilja (MSE), praćene box dijagramom za indikator R^2, za svih 30 izvršavanja. Dijagrami konvergencije funkcije cilja i R^2 za najbolje izvršavanje svakog algoritma prikazani su na Slici 54. Primećuje se da je FA na početku pokazao nešto bržu konvergenciju, međutim,pred kraj izvršavanja, nadmašio ga je predloženi ERSA. Na kraju su na Slici 55 prikazani KDE i swarm plot dijagram.

8.2.2 Wind farm dataset sa EMD dekompozicijom

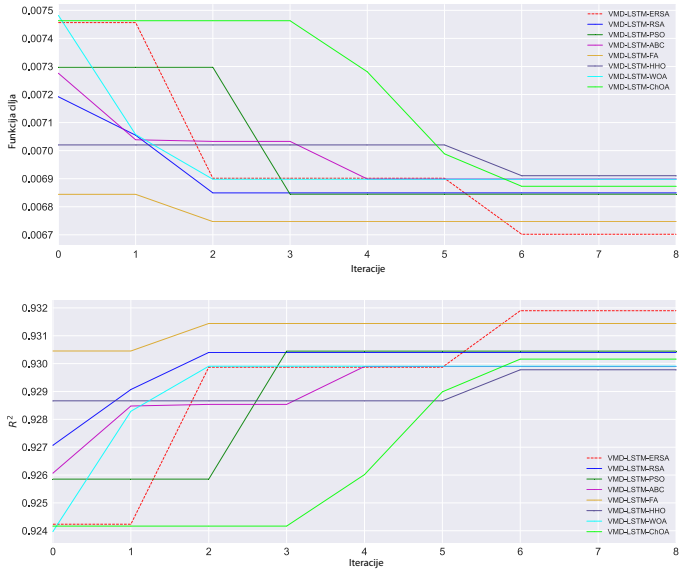
U ovom delu predstavljeni su eksperimentalni rezultate za Wind farm dataset sa primenjenom tehnikom EMD. Tabela 21 prikazuje ukupne metrike u pogledu najboljih, najgorih, srednjih i medijalnih vrednosti, praćenih vrednostima standardne devijacije i varijanse tokom 30 odvojenih izvršavanja svakog posmatranog algoritma.

Tabela 19: *Wind farm dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje sa primenom VMD dekompozicije*

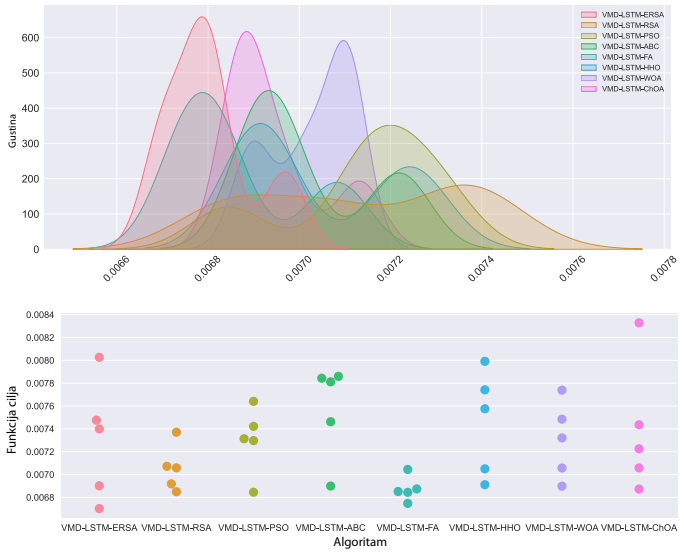
	Indikator geške	VL-ERSA	VL-RSA	VL-PSO	VL-ABC	VL-FA	VL-HHO	VL-WOA	VL-ChOA
Jedan	R^2	0.881572	0.877731	0.885070	0.878542	0.878918	**0.885834**	0.881587	0.883030
korak	MAE	0.075029	0.076310	0.075125	0.076096	0.075810	**0.074311**	0.075867	0.074936
unapred	MSE	0.011655	0.012033	0.011311	0.011953	0.011916	**0.011236**	0.011653	0.011512
	RMSE	0.107959	0.109695	0.106352	0.109331	0.109161	**0.105998**	0.107951	0.107292
Dva	R^2	0.946913	**0.946955**	0.944295	0.946192	0.946887	0.944130	0.944561	0.944182
koraka	MAE	**0.051190**	0.051315	0.052515	0.051676	0.051194	0.052663	0.052508	0.052302
unapred	MSE	0.005224	**0.005220**	0.005482	0.005296	0.005227	0.005498	0.005456	0.005493
	RMSE	0.072281	**0.072253**	0.074041	0.072770	0.072299	0.074151	0.073865	0.074116
Tri	R^2	0.967213	0.966519	0.961981	0.964960	**0.968518**	0.959374	0.963573	0.963274
koraka	MAE	0.041521	0.042333	0.044512	0.042729	**0.040448**	0.046377	0.043806	0.043734
unapred	MSE	0.003226	0.003294	0.003741	0.003447	**0.003097**	0.003997	0.003584	0.003613
	RMSE	0.056802	0.057401	0.061167	0.058721	**0.055660**	0.063230	0.059872	0.060118
Ukupni	R^2	**0.931900**	0.930401	0.930448	0.929897	0.931440	0.929779	0.929906	0.930161
rezultati	MAE	0.055913	0.056652	0.057383	0.056833	**0.055817**	0.057783	0.057393	0.056990
predviđanja	MSE	**0.006701**	0.006848	0.006844	0.006898	0.006746	0.006910	0.006897	0.006872
	RMSE	**0.081865**	0.082760	0.082732	0.083059	0.082140	0.083130	0.083054	0.082903

Tabela 20: *Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra sa primenom VMD dekompozicije*

Metod	Neuroni u sloju 1	Stopa učenja	Epohe	Dropout	Slojevi	Neuroni u sloju 2
VMD-LSTM-ERSA	142	0.009502	600	0.151008	2	127
VMD-LSTM-RSA	140	0.006202	520	0.155388	2	155
VMD-LSTM-PSO	130	0.010000	600	0.200000	2	200
VMD-LSTM-ABC	123	0.008691	529	0.085940	2	152
VMD-LSTM-FA	100	0.010000	600	0.200000	2	100
VMD-LSTM-HHO	200	0.009344	476	0.155241	2	100
VMD-LSTM-WOA	100	0.009352	481	0.094570	2	128
VMD-LSTM-ChOA	144	0.007816	391	0.138489	2	126



Slika 54: *Wind farm dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku sa primenom VMD dekompozicije*



Slika 55: *Wind farm dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku sa primenom VMD dekompozicije*

Tabela 21: *Wind farm dataset - ukupne metrike za najbolje, najgore, srednje vrednosti i medijane, sa primenom EMD dekompozicije*

Metod	Najbolji	Najgori	Srednji	Medijana	Std	Var
EMD-LSTM-ERSA	**0.020351**	0.020556	**0.020460**	0.020556	1.02E-04	1.05E-08
EMD-LSTM-RSA	0.020470	0.020556	0.020513	**0.020513**	4.31E-05	1.86E-09
EMD-LSTM-PSO	0.020491	0.020674	0.020589	0.020674	9.17E-05	8.41E-09
EMD-LSTM-ABC	0.020828	0.021115	0.020971	0.020971	1.44E-04	2.07E-08
EMD-LSTM-FA	0.020512	**0.020549**	0.020531	0.020531	**1.84E-05**	**3.37E-10**
EMD-LSTM-HHO	0.020842	0.020887	0.020860	0.020842	2.17E-05	4.72E-10
EMD-LSTM-WOA	0.020617	0.020666	0.020640	0.020617	2.41E-05	5.78E-10
EMD-LSTM-ChOA	0.020509	0.020935	0.020722	0.020722	2.13E-04	4.53E-08

EMD-LSTM-ERSA model postigao je najbolje rezultate u pogledu najboljih i srednjih vrednosti. Drugi najbolji rezultat postignut je pomoću metode EMD-LSTM-RSA, dok je EMD-LSTM-PSO zauzeo treće mesto. Inače, EMD-LSTM-FA postigao je najbolje rezultate za najgore metrike, kao i za standardnu devijaciju i varijansu. Konačno, najbolja vrednost medijane postignuta je pomoću metode EMD-LSTM-RSA.

U Tabeli 22 prikazane su detaljne metrike za svaki korak predviđanja sa fokusom na najbolje izvršavanje svakog algoritma. Prefiks EL koristi se da označi da je korišćena metoda EMD-LSTM-LSMT. Primećuje se da je predložena EMD-LSTM-ERSA postigla najbolje rezultate za sveukupne rezultate, u pogledu funkcije cilja - MSE, ali i za druge važne indikatore, kao što su R^2 i RMSE (izuzev MAE). Takođe, EMD-LSTM-ERSA postigla je najbolje rezultate za predviđanja za dva koraka unapred, za sve posmatrane indikatore. Najbolje rezultate za predviđanja sa jednim korakom unapred postignuti su pomoću metode EMD-LSTM-FA, dok su najbolji rezultati za predviđanja sa tri koraka unapred postignuti pomoću EMD-LSTM-ChOA. Posmatrajući ukupne rezultate, drugi najbolji algoritam bio je EMD-LSTM-RSA, ispred EMD-LSTM-PSO. Predložena EMD-LSTM-ERSA postigla je najbolju ukupnu vrednost MSE od 0,020350, ispred EMD-LSTM-RSA sa vrednošću MSE od 0.020469.

Najbolji skup LSTM parametara dobijen za najbolje izvršavanje, svake meta-heuristike prikazan je u Tabeli 23. LSTM parametri dobijeni pomoću predloženog EMD-LSTM-ERSA su sledeći: 100 neurona u početnom sloju, stopa učenja od 0,010000, 600 epoha, vrednost odbacivanja od 0,200000, tri sloja i 100 neurona u drugom sloju za slučaj kada je primenjena EMD dekompozicija.

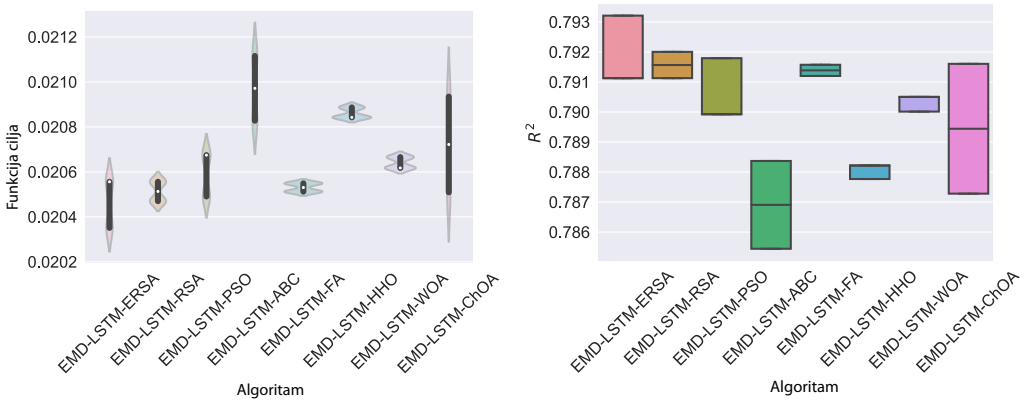
S ciljem pružanja boljeg vizuelnog uvida u rezultate, prikazani su dijagrami na Slikama od 56 - 58. Slika 56 prikazuje violin dijagrame za funkciju cilja (MSE), uz prateći box dijagram za indikator R^2, za svih 30 izvršavanja. Dijagrami konvergencije

Tabela 22: *Wind farm dataset - detaljne metrike svakog koraka predviđanja za najbolje izvršavanje sa primenom EMD dekompozicije*

	Indikator geške	EL-ERSA	EL-RSA	EL-PSO	EL-ABC	EL-FA	EL-HHO	EL-WOA	EL-ChOA
Jedan	R^2	0.770306	0.770573	0.770635	0.766354	**0.770850**	0.766029	0.769614	0.767667
korak	MAE	**0.102287**	0.102416	0.102454	0.103185	0.102444	0.103351	0.102323	0.102841
unapred	MSE	0.022605	0.022579	0.022573	0.022994	**0.022552**	0.023026	0.022673	0.022865
	RMSE	0.150350	0.150263	0.150242	0.151638	**0.150172**	0.151743	0.150576	0.151211
Dva	R^2	**0.799361**	0.798019	0.797192	0.793547	0.796900	0.793721	0.796892	0.797148
koraka	MAE	**0.098350**	0.098588	0.099168	0.099906	0.098911	0.099839	0.098928	0.099220
unapred	MSE	**0.019746**	0.019878	0.019959	0.020318	0.019988	0.020301	0.019989	0.019964
	RMSE	**0.140520**	0.140989	0.141277	0.142541	0.141379	0.142481	0.141381	0.141292
Tri	R^2	0.809964	0.807416	0.807551	0.805203	0.806968	0.804903	0.805001	**0.809996**
koraka	MAE	0.099985	0.099553	0.099394	0.100817	0.099914	0.100610	0.100618	**0.099235**
unapred	MSE	0.018701	0.018952	0.018939	0.019170	0.018996	0.019199	0.019190	**0.018698**
	RMSE	0.136755	0.137669	0.137620	0.138457	0.137828	0.138564	0.138529	**0.136743**
Ukupni	R^2	**0.793210**	0.792002	0.791792	0.788367	0.791572	0.788217	0.790502	0.791603
rezultati	MAE	0.100207	**0.100185**	0.100338	0.101302	0.100422	0.101266	0.100622	0.100431
predviđanja	MSE	**0.020350**	0.020469	0.020490	0.020827	0.020511	0.020841	0.020616	0.020508
	RMSE	**0.142656**	0.143072	0.143144	0.144317	0.143220	0.144368	0.143587	0.143209

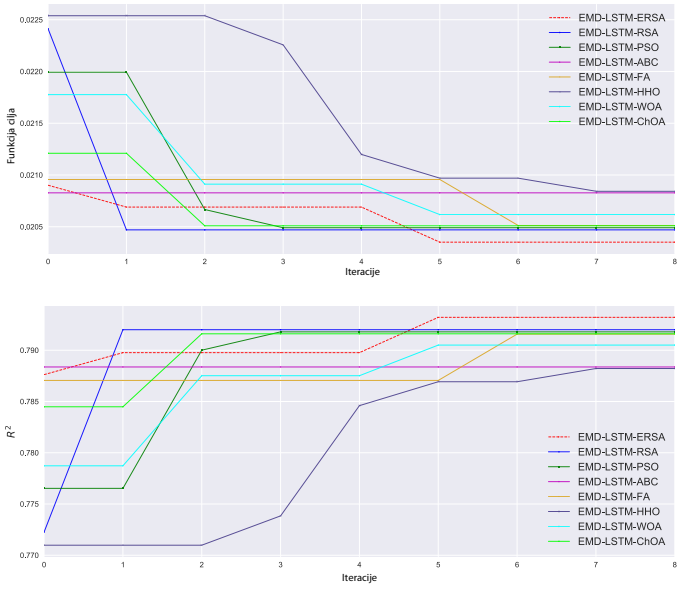
Tabela 23: *Parametri odabrani metaheuristikama za najbolje modele za prognozu vetra sa primenom EMD dekompozicije*

Metod	Neuroni u sloju 1	Stopa učenja	Epohe	Dropout	Slojevi	Neuroni u sloju 2
EMD-LSTM-ERSA	100	0.010000	600	0.200000	3	100
EMD-LSTM-RSA	118	0.005365	600	0.121341	2	181
EMD-LSTM-PSO	100	0.004172	600	0.130481	2	100
EMD-LSTM-ABC	115	0.007103	553	0.153556	2	273
EMD-LSTM-FA	100	0.004116	600	0.126646	2	218
EMD-LSTM-HHO	132	0.003038	600	0.107516	2	259
EMD-LSTM-WOA	125	0.003430	600	0.126450	2	112
EMD-LSTM-ChOA	100	0.005069	600	0.161137	3	234

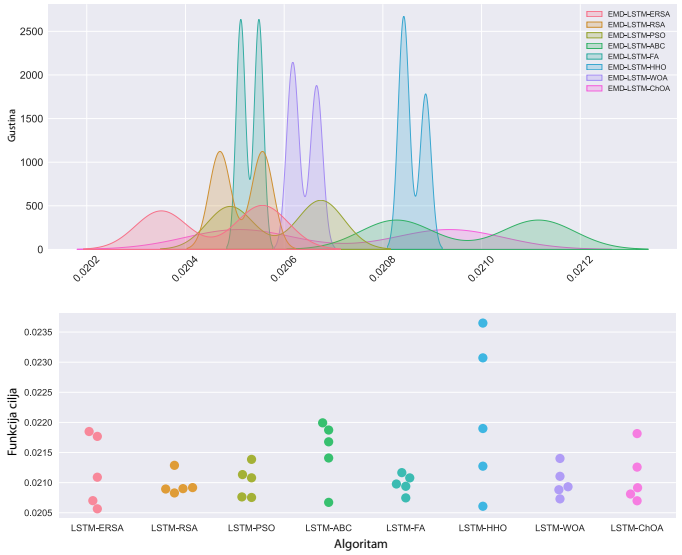


Slika 56: *Wind farm dataset - dijagrami raspodele funkcije cilja i R^2 za svaku metaheuristiku sa primenom EMD dekompozicije*

funkcije cilja i R^2 za najbolje izvršavanje svakog algoritma prikazani su na Slici 57. Može se primetiti da su na početku ChOA, PSO i RSA pokazale nešto bržu konvergenciju, međutim, sve je nadmašio predloženi ERSA u poslednjim rundama izvršavanja. Na kraju, prikazani su KDE i swarm dijagrami na Slici 58. U ovom slučaju, primećuje se da su sva rešenja ERSA grupisana blizu najboljeg rešenja na kraju izvršavanja.



Slika 57: *Wind farm dataset - dijagrami konvergencije funkcije cilja i R^2 vrednosti za svaku metaheuristiku sa primenom EMD dekompozicije*



Slika 58: *Wind farm dataset - KDE dijagram i swarm dijagram funkcije cilja za svaku metaheuristiku sa primenom EMD dekompozicije*

8.2.3 Poređenje sa drugim modelima za Wind farm dataset

Da bi se prikazala komparativna poboljšanja performansi optimalizatora sa tehnikama dekompozicije u uvedenoj metodologiji, najbolji rezultati svakog pristupa upoređeni su sa nekoliko savremenih modela predviđanja. Rezultati funkcije cilja (MSE) i indikatora R^2 prikazani su u Tabeli 24.

Tabela 24: *Poređenje najboljeg modela sa savremenim modelima za predikciju koji su primenjeni na Wind farm dataset*

Metod	MSE	R^2
SVM	0.029070	0.770072
Random forest	0.032550	0.822352
XGBoost	0.023833	0.789320
KELM	0.023340	0.779011
AdaBoost	0.022050	0.775031
ANN (3 Layers)	0.027250	0.766072
RNN (3 Layers)	0.024250	0.787320
LSTM	0.021339	0.783321
LSTM-ERSA	0.020565	0.791029
VMD-LSTM-ERSA	**0.006701**	**0.931900**
EMD-LSTM-ERSA	0.020350	0.793210

Kao što se može primetiti u Tabeli 24, VMD je pokazao najbolje performanse kada je primenjen zajedno sa LSTM modelima koji su optimizovani metaheuristikama.

8.3 Validacija - statistički testovi

Savremena istraživanja u oblasti računarstva zahtevaju od naučnika da utvrde da li su uvedena poboljšanja statistički značajna, jer eksperimentalni rezultati sami po sebi, obično nisu dovoljni da bi se proglasilo da jedan algoritam nadmašuje svoje konkurente. U ovom istraživanju, testirano je osam metoda, uključujući predloženu ERSA metaheuristiku, za podešavanje LSTM mreže, na dva skupa podataka. Izvršeno je poređenje osam metoda primenjenih na 6 različitih problema korišćenih za analizu, kao što je to urađeno u radu [40].

U literaturi koja je citirana u istraživanjima [40, 32] sugeriše se da bi statistički testovi trebalo da obuhvate formiranje reprezentativnog skupa rezultata za svaku metodu, pri čemu se uzorak rezultata dobija određivanjem prosečnih vrednosti funkcija cilja tokom nekoliko nezavisnih izvršavanja za svaki problem. Međutim, ova metodologija može biti manje idealna kada se suočavamo sa raspodelom koja nije normalna. Otvoreno pitanje ostaje da li je korišćenje srednje vrednosti funkcije

Tabela 25: *Rezultati Shapiro-Wilk testa za analizu pojedinačnih problema*

Problem	ERSA	RSA	PSO	ABC	FA	HHO	WOA	ChOA
Wind	0.024	0.026	0.017	0.021	0.029	0.035	0.031	0.039
Wind VMD	0.029	0.030	0.019	0.011	0.042	0.039	0.027	0.034
Wind EMD	0.014	0.022	0.028	0.018	0.033	0.035	0.039	0.036
Wind farm	0.016	0.024	0.027	0.018	0.023	0.028	0.029	0.033
Wind farm VMD	0.023	0.018	0.014	0.024	0.017	0.031	0.034	0.028
Wind farm EMD	0.021	0.015	0.023	0.019	0.031	0.033	0.031	0.022

Tabela 26: *Rezultati Shapiro-Wilk testa za analizu višestrukih problema*

Metod	ERSA	RSA	PSO	ABC	FA	HHO	WOA	ChOA
$p - vrednost$	0.0084	0.0093	0.0061	0.0087	0.0074	0.0095	0.0132	0.0147

cilja adekvatno za statističko poređenje stohastičkih metoda, kako je navedeno u pregledu literature citirane u radu [40]. Bez obzira na ove potencijalne nedostatke, stopa greške klasifikacije funkcije cilja je usrednjena tokom 30 nezavisnih izvršavanja kako bi se uporedilo 10 metoda preko 6 različitih problema u ovom istraživanju.

Odluka je doneta nakon sprovođenja Shapiro-Wilk testa [111] za analizu pojedinačnih problema, pri čemu je korišćen sledeći postupak: uzorak podataka je formiran za svaki algoritam i svaki problem prikupljanjem rezultata za svako izvršavanje, a odgovarajuće p-vrednosti su izračunate za sve kombinacije metoda i problema. Rezultirajuće p-vrednosti prikazane su u Tabeli 25.

S obzirom da su p-vrednosti u Tabeli 25 sve manje od $\alpha = 0.05$, što znači da nisu rezultat slučajnosti, nulta hipoteza može biti odbačena. Kao rezultat toga, uzorci podataka za sve kombinacije metoda i problema ne potiču iz Gausove raspodele, što znači da nije prihvatljivo koristiti prosečnu vrednost funkcije cilja u daljim statističkim testovima.

Zatim je sprovedena statistička analiza više problema i više metoda, koristeći uzorke podataka dobijene za najbolje vrednosti funkcije cilja, za više od 30 pokretanja svakog pojedinačnog algoritma, za svaki problem. Kako bi se osigurala validna upotreba parametarskih testova, morali su biti ispunjeni sledeći uslovi: nezavisnost, normalnost i jednakost varijansi podataka, kako je opisano u radu [74]. Nezavisnost je obezbeđena na taj način što je svako izvršavanje započeto generisanjem skupa nasumičnih rešenja. Da bi se procenilo da li se radi o normalnoj raspodeli, uzorci podataka su podvrgnuti Shapiro-Wilk testu, a rezultati za svaki algoritam su prikazani u Tabeli 26.

Iako je sproveden Shapiro-Wilk testa za sve metode, dobijene rezultirajuće p-

Tabela 27: *Rezultati Friedman testa*

Metodi	ERSA	RSA	PSO	ABC	FA	HHO	WOA	ChOA
Wind	1	5	7	2	6	8	3	4
Wind VMD	1	3	6	8	4	5	7	2
Wind EMD	1	7	2	8	3	4	6	5
Wind Farm	1	8	7	3	6	2	5	4
Wind Farm VMD	1	4	3	7	2	8	6	5
Wind Farm EMD	1	2	3	7	5	8	6	4
Prosečno rang	1.00	4.83	4.67	5.83	4.33	5.83	5.50	4.00
Rang	1	5	4	7	3	8	6	2

Tabela 28: *Rezultati Friedman aligned testa*

Metodi	ERSA	RSA	PSO	ABC	FA	HHO	WOA	ChOA
Wind	14	22	29	18	26	30	20	21
Wind VMD	12	16	28	34	17	27	31	15
Wind EMD	1	43	9	45	19	37	40	38
Wind Farm	4	46	39	13	36	7	32	23
Wind Farm VMD	3	25	24	42	6	44	41	33
Wind Farm EMD	2	5	8	47	11	48	35	10
Prosečni rang	6.00	26.17	22.83	33.17	19.17	32.17	33.17	23.33
Rang	1	5	3	7	2	6	8	4

vrednosti su značajno manje od $\alpha = 0.05$, kako je naznačeno u Tabeli 26. Ovo ukazuje da pretpostavka opravdane upotrebe parametarskih testova nije ispunjena, pa su umesto toga korišćeni neparametarski testovi. Predloženi ERSA je izabran kao kontrolni algoritam za sve neparametarske testove.

Friedman test [44, 45] sa dvosmernom analizom varijanse uz rangiranje, primenjen je kako bi se utvrdilo da li je nivo performansi predloženog ERSA značajno bolji u odnosu na druge algoritme. Ovaj test je primenjen zajedno sa Holm post-hoc procedurom, koju su predložili [32]. Rezultati Friedman testa prikazani su u Tabeli 27, uz rezultate Friedman-aligned testa prikazanih u Tabeli 28.

Rezultati prikazani u Tabeli 27 pokazuju da je predloženi ERSA metod postigao superiorni nivo performansi u odnosu na ostale metode u uporednoj analizi, sa prosečnim rangom 1.00. Drugi najbolji rezultat postignut je pomoću ChOA algoritma sa prosečnim rangom od 4.00, dok je FA na trećem mestu sa prosečnim rangom od 4.33. RSA algoritam postigao je prosečni rang od 4.83, što pokazuje očiglednu superiornost predloženog ERSA u odnosu na osnovnu verziju algoritma. Osim toga, Friedman-ova statistika ($\chi_r^2 = 17.22$) veća je od χ^2 kritične vrednosti sa sedam stepeni slobode (14.07) na nivou značajnosti $\alpha = 0.05$. Kako je Friedman p-vrednost 1.24×10^{-9}, zaključuje se da se rezultati dobijeni posmatranim metodama međusobno značajno razlikuju. Konačno, to omogućava odbacivanje nulte hipoteze (H_0) i potvrđuje da je

Tabela 29: *Holm's step-down procedura*

Poređenje	p-vrednosti	Rangiranje	$\alpha = 0.05$	$\alpha = 0.1$	H1	H2
ERSA - ABC	0.000315783359126	0	0.007143	0.014286	1	1
ERSA - HHO	0.000315783359126	1	0.008333	0.016667	1	1
ERSA - WOA	0.000731358293341	2	0.010000	0.020000	1	1
ERSA - RSA	0.00335832275131	3	0.012500	0.025000	1	1
ERSA - PSO	0.004760945592049	4	0.016667	0.033333	1	1
ERSA - FA	0.00921106272705	5	0.025000	0.050000	1	1
ERSA - ChOA	0.016947426762345	6	0.050000	0.100000	1	1

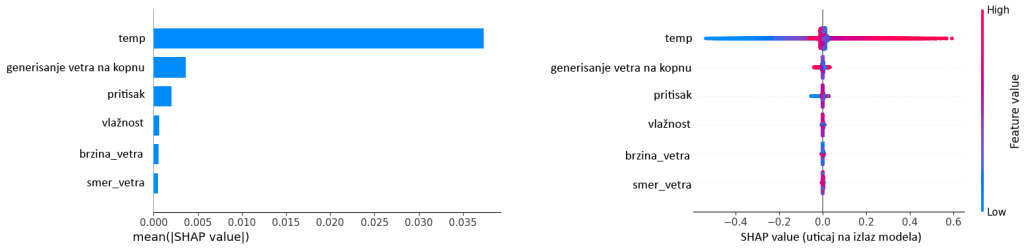
predloženi ERSA metod značajno statistički različit od drugih konkurenata. Slični zaključci proizilaze i iz rezultata dobijenih Friedman-aligned testom, koji su dati u Tabeli 28.

Dodatno je sproveden i Iman i Davenport-ov test [59]. Pomoću Iman i Davenport-ovog testa dobijena je vrednost 3.47, koja je veća od kritične vrednosti F-distribucije 2.28. Iz čega se može zaključiti, da ovaj poseban test takođe odbacuje H_0 hipotezu.

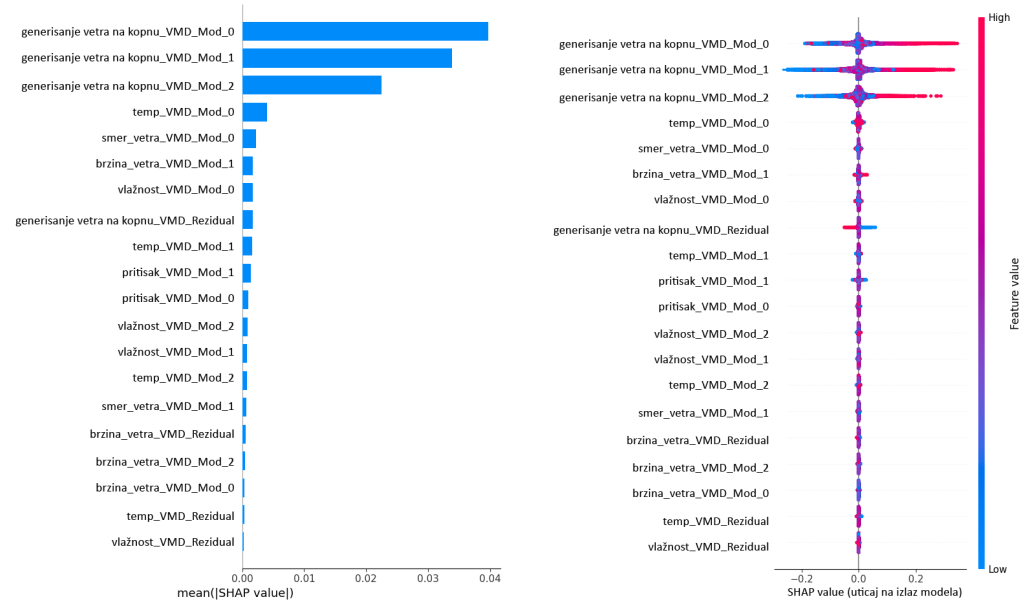
Budući da oba sprovedena testa odbacuju nultu hipotezu, korišćena je neparametarska post-hoc Holm-ova procedura, čiji su rezultati prikazani u Tabeli 29. Ova procedura sortira posmatrane metode na osnovu p-vrednosti, koje su procenjene na osnovu $\alpha/(k-i)$, gde k predstavljaju stepen slobode ($k = 7$ u ovom istraživanju), i je broj metoda, nakon sortiranja u rastućem nizu (u odnosu na rang). U ovom istraživanju korišćene su vrednosti α 0.05 i 0.1. Rezultati prikazani u Tabeli 29 jasno ukazuju da je predstavljeni ERSA metod značajno statistički nadmašio sve konkurente za oba razmatrana nivoa značajnosti 0.05 i 0.1.

8.4 Interpretacija najboljeg modela i SHAP analiza

Tumačenje najboljeg modela za oba posmatrana skupa podataka izvršeno je i pomoću SHAP analize. Za procenu značaja pojedinih karakteristika korišćene su SHAP vrednosti. Ove vrednosti upoređuju modele sa i bez svake pojedinačne karakteristike, i na taj način prikazuju uticaj određene karakteristike na izlaz posmatranog modela. Pošto redosled karakteristika može uticati na predviđanje, značaj karakteristika se računa u svakom mogućem redosledu kako bi se osiguralo korektno poređenje karakteristika [48].



Slika 59: *Uticaj karakteristika na najbolji LSTM model za Wind dataset bez dekompozicije, određen SHAP analizom*

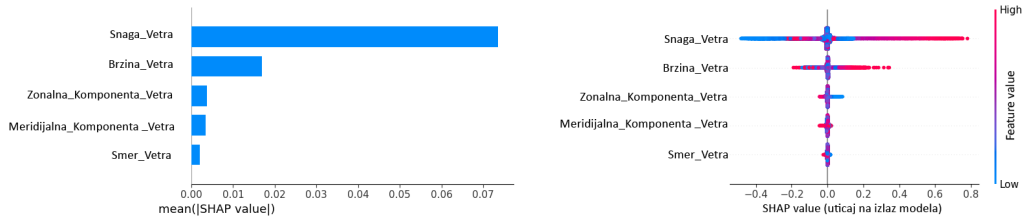


Slika 60: *Uticaj karakteristika na najbolji LSTM model za Wind dataset sa VMD dekompozicijom, određen SHAP analizom*

8.4.1 Wind dataset SHAP analiza

Na Slici 59 prikazani su uticaji svake karakteristike za model LSTM-ERSA za Wind dataset, dok Slika 60 prikazuje uticaje najboljeg modela sa VMD dekompozicijom, odnosno VMD-LSTM-ERSA. Ukoliko se detaljnije razmatra dijagram na levoj strani slike, može se zaključiti da je najvažnija karakteristika u ovom slučaju temperatura, zatim generisanje vetra na kopnu i pritisak.

Može se primetiti da se uticaj karakteristika značajno menja nakon primene VMD tenike dekompozicije. Model koji je postigao najbolje rezultate bez dekompozicije, najveću vrednost je dodelio temperaturi. Međutim, nakon primene VMD-a, prva tri dekomponovana moda generisanja vetra na kopnu, postaju najznačajnije karakteristike. Ovo je najverovatnije zbog šuma koji se javlja u podacima o generisanju vetra na kopnu. U sirovom obliku, proizvodnja vetra na kopnu je prilično



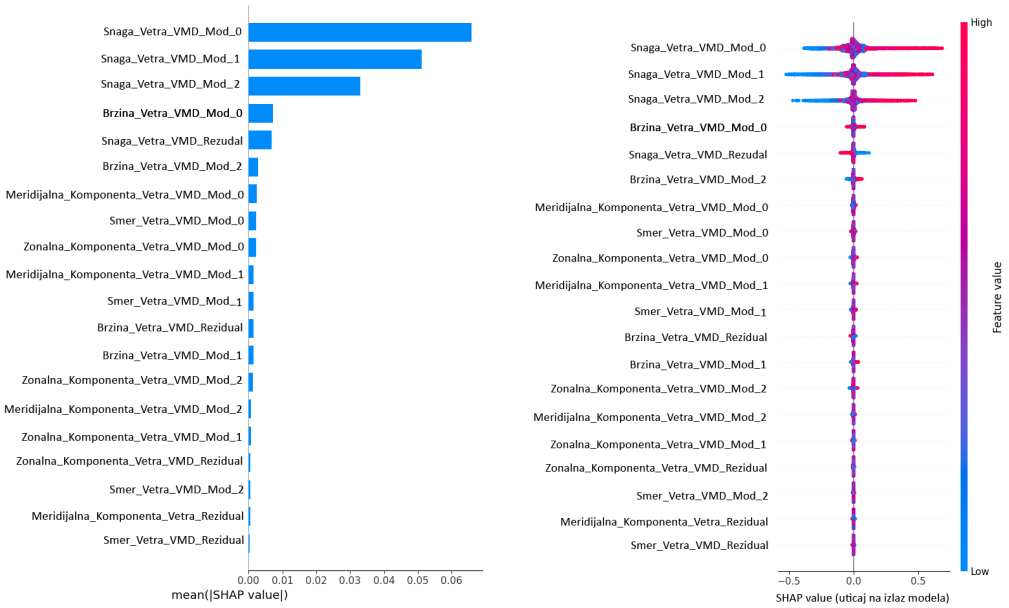
Slika 61: *Uticaj karakteristika na najbolji LSTM model za Wind farm dataset bez dekompozicije, određen SHAP analizom*

složen niz podataka koji sadrži šum, dok je temperatura predvidljiviji niz podataka manje podložan naglim promenama. Nakon dekompozicije, modovi generisanja vetra na kopnu postaju pouzdaniji, predvidljiviji i sa manje šuma, što omogućava ovim karakteristikama da imaju značajniji uticaj na model, veći čak i od temperature. Takođe je interesantno primetiti da rezidualne vrednosti za svaki mod (komponente podataka koje nisu mogle biti dekomponovane u specifičnu modu i uglavnom sadrže šum), imaju najmanji uticaj na predikcije modela.

8.4.2 Wind farm dataset SHAP analiza

Na Slici 61 prikazani su uticaji svake karakteristike za najbolji model LSTM-ERSA za Wind farm dataset. Slika 62 prikazuje uticaj karakteristika nakon dekompozicije podataka pomoću VMD, koji je nazvan VMD-LSTM-ERSA.

Rezultati SHAP analize prikazuju značajan uticaj snage vetra, koji prati brzina vetra. Ovi rezultati su dodatno potvrđeni analizom modela koji su primenjeni na podatke sa dekompozicijom. Snaga vetra, koju prate komponente brzine vetra, pokazuje značajan uticaj. Takođe se primjećuje da je uticaj rezidualnih komponenati, koje uglavnom čine šum, nizak za većinu karakteristika. Međutim, rezidual snage vetra pokazuje značajan uticaj na predviđanja.



Slika 62: *Uticaj karakteristika na najbolji LSTM model za Wind farm dataset sa VMD dekompozicijom, određen SHAP analizom*

9 Zaključak

Na početku ovog istraživanja, predložena je unapređena verzija meteheuristike roja poznate kao RSA, koja prevazilazi poznate nedostatke izvorno predstavljenog algoritma. Ovaj algoritam je nazvan ERSA i korišćen za podešavanje hiperparametara LSTM mreže za probleme proizvodnje energije vetra.

Ova metodologija je procenjena na dva skupa podataka o proizvodnji vetra za tri različita scenarija za svaki skup podataka - bez dekompozicije, sa VMD i sa EMD tehnikom dekompozicije. Predviđanja su izvršena do tri koraka unapred, a rezultati su upoređeni sa rezultatima dobijenim pomoću konkurentnih meteheuristika, pri čemu su korišćeni identične postavke eksperimenta. Dobijeni rezultati jasno ukazuju na superiornost predloženog LSTM-ERSA modela u svim posmatranim slučajevima. Statistička analiza koja je sprovedena, pokazala je da su rezultati dobijeni predloženom metodom statistički značajni. Takođe, jasno je pokazano da odabir odgovarajućih parametara značajno poboljšava performanse optimizovanih mreža. Na kraju je izvedena SHAP analiza za najuspešniji model, za svaki skup podataka, sa ciljem da se proceni uticaj karakteristika na svaki model.

Sprovedeno istraživanje pokazuje veliki potencijal za korišćenje hibridnih ML modela optimizovanih pomoću algoritama meteheuristika, za prognozu proizvodnje energije vetra. Značaj zadatka kao što je procena očekivane proizvodnje električne energije u vetroparku je veliki. Mreža mora biti u stanju ravnoteže između proizvodnje i potrošnje energije u svakom trenutku. Buduća istraživanja ovog važnog zadatka, biće usmerena na razvoj još preciznijih modela optimizovanih različitim metaheurističkim algoritmima. Takođe će se u budućnosti istraživati primena novih algoritama za dekompoziciju i njihova optimizacija. Predložena modifikovana meteheuristika će biti primenjivana na nove izazove.

10 Literatura

[1] Abraham, A., Guo, H., Liu, H.: Swarm intelligence: foundations, perspectives and applications. In: Swarm intelligent systems, pp. 3–25. Springer (2006)

[2] Abualigah, L., Abd Elaziz, M., Sumari, P., Geem, Z.W., Gandomi, A.H.: Reptile search algorithm (rsa): A nature-inspired meta-heuristic optimizer. Expert Systems with Applications 191, 116158 (2022)

[3] Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., Gandomi, A.H.: The arithmetic optimization algorithm. Computer methods in applied mechanics and engineering 376, 113609 (2021)

[4] Aggarwal, C.C., et al.: Neural networks and deep learning. Springer 10(978), 3 (2018)

[5] Akella, A., Saini, R., Sharma, M.P.: Social, economical and environmental impacts of renewable energy systems. Renewable energy 34(2), 390–396 (2009)

[6] Aldahiri, A., Alrashed, B., Hussain, W.: Trends in using iot with machine learning in health prediction system. Forecasting 3(1), 181–206 (2021), `https://www.mdpi.com/2571-9394/3/1/12`

[7] Amalou, I., Mouhni, N., Abdali, A.: Multivariate time series prediction by rnn architectures for energy consumption forecasting. Energy Reports 8, 1084–1091 (2022)

[8] Andonie, R.: Hyperparameter optimization in learning systems. Journal of Membrane Computing 1(4), 279–291 (2019)

[9] Arora, R.K.: Optimization: algorithms and applications. CRC press (2015)

[10] Awerbuch, S., Berger, M.: Applying portfolio theory to eu electricity planning and policy-making (2003)

[11] Ayodele, T.O.: Types of machine learning algorithms. New advances in machine learning 3, 19–48 (2010)

[12] Bacanin, N., Bezdan, T., Tuba, E., Strumberger, I., Tuba, M., Zivkovic, M.: Task scheduling in cloud computing environment by grey wolf optimizer. In: 2019 27th Telecommunications Forum (TELFOR). pp. 1–4. IEEE (2019)

[13] Bacanin, N., Sarac, M., Budimirovic, N., Zivkovic, M., AlZubi, A.A., Bashir, A.K.: Smart wireless health care system using graph lstm pollution prediction and dragonfly node localization. Sustainable Computing: Informatics and Systems 35, 100711 (2022)

[14] Bacanin, N., Stoean, C., Zivkovic, M., Jovanovic, D., Antonijevic, M., Mladenovic, D.: Multi-swarm algorithm for extreme learning machine optimization. Sensors 22(11), 4204 (2022)

[15] Bacanin, N., Stoean, C., Zivkovic, M., Rakic, M., Strulak-Wójcikiewicz, R., Stoean, R.: On the benefits of using metaheuristics in the hyperparameter tuning of deep learning models for energy load forecasting. Energies 16(3), 1434 (2023)

[16] Bacanin, N., Venkatachalam, K., Bezdan, T., Zivkovic, M., Abouhawwash, M.: A novel firefly algorithm approach for efficient feature selection with covid-19 dataset. Microprocessors and Microsystems 98, 104778 (2023)

[17] Bacanin, N., Zivkovic, M., Al-Turjman, F., Venkatachalam, K., Trojovskỳ, P., Strumberger, I., Bezdan, T.: Hybridized sine cosine algorithm with convolutional neural networks dropout regularization application. Scientific Reports 12(1), 1–20 (2022)

[18] Bacanin, N., Zivkovic, M., Stoean, C., Antonijevic, M., Janicijevic, S., Sarac, M., Strumberger, I.: Application of natural language processing and machine learning boosted with swarm intelligence for spam email filtering. Mathematics 10(22), 4173 (2022)

[19] Barika, R., Faust, O.: A review of automated sleep stage scoring. In: Reference Module in Neuroscience and Biobehavioral Psychology. Elsevier (2021)

[20] Belotti, J., Siqueira, H., Araujo, L., Stevan Jr, S.L., de Mattos Neto, P.S., Marinho, M.H., de Oliveira, J.F.L., Usberti, F., Leone Filho, M.d.A., Converti, A.,

et al.: Neural-based ensembles and unorganized machines to predict streamflow series from hydroelectric plants. Energies 13(18), 4769 (2020)

[21] Beni, G.: Swarm intelligence. Complex Social and Behavioral Systems: Game Theory and Agent-Based Models pp. 791–818 (2020)

[22] Bezdan, T., Cvetnic, D., Gajic, L., Zivkovic, M., Strumberger, I., Bacanin, N.: Feature selection by firefly algorithm with improved initialization strategy. In: 7th Conference on the Engineering of Computer Based Systems. pp. 1–8 (2021)

[23] Bezdan, T., Zivkovic, M., Antonijevic, M., Zivkovic, T., Bacanin, N.: Enhanced flower pollination algorithm for task scheduling in cloud computing environment. In: Machine Learning for Predictive Analysis, pp. 163–171. Springer (2020)

[24] Bezdan, T., Zivkovic, M., Tuba, E., Strumberger, I., Bacanin, N., Tuba, M.: Glioma brain tumor grade classification from mri using convolutional neural networks designed by modified fa. In: International Conference on Intelligent and Fuzzy Systems. pp. 955–963. Springer (2020)

[25] Bezdan, T., Zivkovic, M., Tuba, E., Strumberger, I., Bacanin, N., Tuba, M.: Multi-objective task scheduling in cloud computing environment by hybridized bat algorithm. In: International Conference on Intelligent and Fuzzy Systems. pp. 718–725. Springer (2020)

[26] Bukumira, M., Antonijevic, M., Jovanovic, D., Zivkovic, M., Mladenovic, D., Kunjadic, G.: Carrot grading system using computer vision feature parameters and a cascaded graph convolutional neural network. Journal of Electronic Imaging 31(6), 061815 (2022)

[27] Cheng, S., Shi, Y.: Diversity control in particle swarm optimization. In: 2011 IEEE Symposium on Swarm Intelligence. pp. 1–9. IEEE (2011)

[28] Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp.

1724–1734. Association for Computational Linguistics, Doha, Qatar (Oct 2014), https://aclanthology.org/D14-1179

[29] Coppitters, D., Contino, F.: Optimizing upside variability and antifragility in renewable energy system design. Scientific Reports 13(1), 9138 (2023)

[30] Cornuejols, G., Tutuncu, R.: Optimization methods in finance. Camegie Mellon University, Pittsburgh (2005)

[31] Delahaye, D., Chaimatanan, S., Mongeau, M.: Simulated annealing: From basics to applications. Handbook of metaheuristics pp. 1–35 (2019)

[32] Derrac, J., García, S., Molina, D., Herrera, F.: A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation 1(1), 3–18 (2011)

[33] Devi, A.S., Maragatham, G., Boopathi, K., Rangaraj, A.: Hourly day-ahead wind power forecasting with the eemd-cso-lstm-efg deep learning technique. Soft Computing 24(16), 12391–12411 (2020)

[34] Dorigo, M.: Optimization, learning and natural algorithms. Ph. D. Thesis, Politecnico di Milano (1992)

[35] Dorigo, M., Birattari, M., Stutzle, T.: Ant colony optimization. IEEE computational intelligence magazine 1(4), 28–39 (2006)

[36] Dragomiretskiy, K., Zosso, D.: Variational mode decomposition. IEEE transactions on signal processing 62(3), 531–544 (2013)

[37] Dreyfus, G.: Neural networks: methodology and applications. Springer Science & Business Media (2005)

[38] Du, K.L., Swamy, M., et al.: Search and optimization by metaheuristics. Techniques and Algorithms Inspired by Nature pp. 1–10 (2016)

[39] Dwivedi, R., Dave, D., Naik, H., Singhal, S., Omer, R., Patel, P., Qian, B., Wen, Z., Shah, T., Morgan, G., et al.: Explainable ai (xai): Core ideas, techniques, and solutions. ACM Computing Surveys 55(9), 1–33 (2023)

[40] Eftimov, T., Korošec, P., Seljak, B.K.: Disadvantages of statistical comparison of stochastic optimization algorithms. Proceedings of the Bioinspired Optimizaiton Methods and their Applications, BIOMA pp. 105–118 (2016)

[41] Emmerich, M., Shir, O.M., Wang, H.: Evolution strategies. In: Handbook of Heuristics, pp. 89–119. Springer (2018)

[42] Fan, H., Jiang, M., Xu, L., Zhu, H., Cheng, J., Jiang, J.: Comparison of long short term memory networks and the hydrological model in runoff simulation. Water 12(1), 175 (2020)

[43] Fausto, F., Reyna-Orta, A., Cuevas, E., Andrade, Á.G., Perez-Cisneros, M.: From ants to whales: metaheuristics for all tastes. Artificial Intelligence Review 53(1), 753–810 (2020)

[44] Friedman, M.: The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the american statistical association 32(200), 675–701 (1937)

[45] Friedman, M.: A comparison of alternative tests of significance for the problem of m rankings. The Annals of Mathematical Statistics 11(1), 86–92 (1940)

[46] Fu, J., Chu, J., Guo, P., Chen, Z.: Condition monitoring of wind turbine gearbox bearing based on deep learning model. Ieee Access 7, 57078–57087 (2019)

[47] Gajic, L., Cvetnic, D., Zivkovic, M., Bezdan, T., Bacanin, N., Milosevic, S.: Multi-layer perceptron training using hybridized bat algorithm. In: Computational Vision and Bio-Inspired Computing, pp. 689–705. Springer (2021)

[48] García, M.V., Aznarte, J.L.: Shapley additive explanations for no2 forecasting. Ecological Informatics 56, 101039 (2020)

[49] Gaytán Cortés, J.: Financial and economic indicators. the monte carlo method of random simulation samples. Mercados y negocios 24(50), 95–107 (2023)

[50] Gen, M., Cheng, R.: Genetic algorithms and engineering optimization, vol. 7. John Wiley & Sons (1999)

[51] Glover, F.: Future paths for integer programming and links to artificial intelligence. Computers & operations research 13(5), 533–549 (1986)

[52] Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT press (2016)

[53] Goss, S., Aron, S., Deneubourg, J.L., Pasteels, J.M.: Self-organized shortcuts in the argentine ant. Naturwissenschaften 76(12), 579–581 (1989)

[54] Heidari, A.A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., Chen, H.: Harris hawks optimization: Algorithm and applications. Future generation computer systems 97, 849–872 (2019)

[55] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9(8), 1735–1780 (1997)

[56] Hong, T., Pinson, P., Fan, S.: Global energy forecasting competition 2012. International Journal of Forecasting 30(2), 357–363 (2014), https://www.sciencedirect.com/science/article/pii/S0169207013000745

[57] Huang, N.E., Shen, Z., Long, S.R., Wu, M.C., Shih, H.H., Zheng, Q., Yen, N.C., Tung, C.C., Liu, H.H.: The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. Proceedings of the Royal Society of London. Series A: mathematical, physical and engineering sciences 454(1971), 903–995 (1998)

[58] IEA: International energy agency, https://www.iea.org/

[59] Iman, R.L., Davenport, J.M.: Approximations of the critical region of the fbietkan statistic. Communications in Statistics-Theory and Methods 9(6), 571–595 (1980)

[60] IRENA: International renewable energy agency, https://www.irena.org/

[61] Jiang, W., Ren, Y., Liu, Y., Leng, J.: Artificial neural networks and deep learning techniques applied to radar target detection: a review. Electronics 11(1), 156 (2022)

[62] Jiao, M., Wang, D., Qiu, J.: A gru-rnn based momentum optimized algorithm for soc estimation. Journal of Power Sources 459, 228051 (2020)

[63] Jovanovic, D., Antonijevic, M., Stankovic, M., Zivkovic, M., Tanaskovic, M., Bacanin, N.: Tuning machine learning models using a group search firefly algorithm for credit card fraud detection. Mathematics 10(13), 2272 (2022)

[64] Jovanovic, L., Djuric, M., Zivkovic, M., Jovanovic, D., Strumberger, I., Antonijevic, M., Budimirovic, N., Bacanin, N.: Tuning xgboost by planet optimization algorithm: An application for diabetes classification. In: Proceedings of Fourth International Conference on Communication, Computing and Electronics Systems: ICCCES 2022. pp. 787–803. Springer (2023)

[65] Jovanovic, L., Jovanovic, D., Bacanin, N., Jovancai Stakic, A., Antonijevic, M., Magd, H., Thirumalaisamy, R., Zivkovic, M.: Multi-step crude oil price prediction based on lstm approach tuned by salp swarm algorithm with disputation operator. Sustainability 14(21), 14616 (2022)

[66] Jovanovic, L., Jovanovic, G., Perisic, M., Alimpic, F., Stanisic, S., Bacanin, N., Zivkovic, M., Stojic, A.: The explainable potential of coupling metaheuristics-optimized-xgboost and shap in revealing vocs' environmental fate. Atmosphere 14(1), 109 (2023)

[67] Karaboga, D., Basturk, B.: On the performance of artificial bee colony (abc) algorithm. Applied Soft Computing 8(1), 687–697 (2008), https://www.sciencedirect.com/science/article/pii/S1568494607000531

[68] Karaboga, D., Basturk, B.: On the performance of artificial bee colony (ABC) algorithm. Applied soft computing 8(1), 687–697 (2008)

[69] Karaboğa, D.: An idea based on honey bee swarm for numerical optimization (2005), https://api.semanticscholar.org/CorpusID:8215393

[70] Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks. vol. 4, pp. 1942–1948. IEEE (1995)

[71] Khishe, M., Mosavi, M.R.: Chimp optimization algorithm. Expert systems with applications 149, 113338 (2020)

[72] Kirkpatrick, S., Gelatt Jr, C.D., Vecchi, M.P.: Optimization by simulated annealing. science 220(4598), 671–680 (1983)

[73] Landau, D.P., Binder, K.: A Guide to Monte Carlo Simulationsin Statistical Physics Third Edition (2009)

[74] LaTorre, A., Molina, D., Osaba, E., Poyatos, J., Del Ser, J., Herrera, F.: A prescription of methodological guidelines for comparing bio-inspired optimization algorithms. Swarm and Evolutionary Computation 67, 100973 (2021)

[75] Laughton, M.A.: Renewable energy sources: Watt committee: Report number 22. CRC Press (1990)

[76] Li, M., Yao, J., Shen, Y., Yuan, B., Simmonds, I., Liu, Y.: Impact of synoptic circulation patterns on renewable energy-related variables over china. Renewable Energy (2023)

[77] Liu, B., Zhao, S., Yu, X., Zhang, L., Wang, Q.: A novel deep learning approach for wind power forecasting based on wd-lstm model. Energies 13(18), 4964 (2020)

[78] Liu, Y., Guan, L., Hou, C., Han, H., Liu, Z., Sun, Y., Zheng, M.: Wind power short-term prediction based on lstm and discrete wavelet transform. Applied Sciences 9(6), 1108 (2019)

[79] Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 4765–4774. Curran Associates, Inc. (2017), http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

[80] Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. Advances in neural information processing systems 30 (2017)

[81] Manwell, J.F., McGowan, J.G., Rogers, A.L.: Wind energy explained: theory, design and application. John Wiley & Sons (2010)

[82] Marsland, S.: Machine learning: an algorithmic perspective. Chapman and Hall/CRC (2011)

[83] de Mattos Neto, P.S., de Oliveira, J.F., Júnior, D.S.d.O.S., Siqueira, H.V., Marinho, M.H., Madeiro, F.: An adaptive hybrid system using deep learning for wind speed forecasting. Information Sciences 581, 495–514 (2021)

[84] McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics 5, 115–133 (1943)

[85] Meignan, D., Knust, S., Frayret, J.M., Pesant, G., Gaud, N.: A review and taxonomy of interactive optimization methods in operations research. ACM Transactions on Interactive Intelligent Systems (TiiS) 5(3), 1–43 (2015)

[86] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculations by fast computing machines. The journal of chemical physics 21(6), 1087–1092 (1953)

[87] Metropolis, N., Ulam, S.: The monte carlo method. Journal of the American statistical association 44(247), 335–341 (1949)

[88] Milosevic, S., Bezdan, T., Zivkovic, M., Bacanin, N., Strumberger, I., Tuba, M.: Feed-forward neural network training by hybrid bat algorithm. In: Modelling and Development of Intelligent Systems: 7th International Conference, MDIS 2020, Sibiu, Romania, October 22–24, 2020, Revised Selected Papers 7. pp. 52–66. Springer International Publishing (2021)

[89] Mirjalili, S.: Sca: a sine cosine algorithm for solving optimization problems. Knowledge-based systems 96, 120–133 (2016)

[90] Mirjalili, S., Lewis, A.: The whale optimization algorithm. Advances in engineering software 95, 51–67 (2016)

[91] Muhuri, P.S., Chatterjee, P., Yuan, X., Roy, K., Esterline, A.: Using a long short-term memory recurrent neural network (lstm-rnn) to classify network attacks. Information 11(243), 5 (2020)

[92] Nayak, S.: Fundamentals of optimization techniques with algorithms. Academic Press (2020)

[93] Nocedal, J., Wright, S.J.: Numerical optimization. Springer (1999)

[94] Pavlov-Kagadejev, M., Jovanovic, L., Bacanin, N., Deveci, M., Zivkovic, M., Tuba, M., Strumberger, I., Pedrycz, W.: Optimizing long-short-term memory models via metaheuristics for decomposition aided wind energy generation forecasting. Artificial Intelligence Review 57(3), 45 (2024)

[95] Pengelly, J.: Monte carlo methods. University of Otago (2002)

[96] Petrovic, A., Bacanin, N., Zivkovic, M., Marjanovic, M., Antonijevic, M., Strumberger, I.: The adaboost approach tuned by firefly metaheuristics for fraud detection. In: 2022 IEEE World Conference on Applied Intelligence and Computing (AIC). pp. 834–839. IEEE (2022)

[97] Preuss, M., Stoean, C., Stoean, R.: Niching foundations: Basin identification on fixed-property generated landscapes. In: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation. p. 837–844. GECCO '11, Association for Computing Machinery, New York, NY, USA (2011)

[98] Probst, P., Boulesteix, A.L., Bischl, B.: Tunability: Importance of hyperparameters of machine learning algorithms. The Journal of Machine Learning Research 20(1), 1934–1965 (2019)

[99] Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.A.: Quasi-oppositional differential evolution. In: 2007 IEEE Congress on Evolutionary Computation. pp. 2229–2236 (2007)

[100] Rangaiah, G.P., Petriciolet, A.: Multi-objective optimization in chemical engineering. Developments and applications/edited by Gade Pandu Rangaiah, Adrián Bonilla-Petriciolet (2013)

[101] Razmjoo, A., Kaigutha, L.G., Rad, M.V., Marzband, M., Davarpanah, A., Denai, M.: A technical analysis investigating energy sustainability utilizing reliable renewable energy sources to reduce co2 emissions in a high potential area. Renewable Energy 164, 46–57 (2021)

[102] ur Rehman, N., Aftab, H.: Multivariate variational mode decomposition. IEEE Transactions on signal processing 67(23), 6039–6052 (2019)

[103] Rehman, N., Mandic, D.P.: Multivariate empirical mode decomposition. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 466(2117), 1291–1302 (2010)

[104] Salb, M., Jovanovic, L., Zivkovic, M., Tuba, E., Elsadai, A., Bacanin, N.: Training logistic regression model by enhanced moth flame optimizer for spam email classification pp. 753–768 (2022)

[105] Sangeetha, J., Kumaran, U.: A hybrid optimization algorithm using bilstm structure for sentiment analysis. Measurement: Sensors p. 100619 (2022)

[106] Sarker, I.H.: Machine learning: Algorithms, real-world applications and research directions. SN computer science 2(3), 160 (2021)

[107] Sathaye, J., Meyers, S., Sathaye, J., Meyers, S.: Renewable energy supply. Greenhouse Gas Mitigation Assessment: A Guidebook pp. 133–149 (1995)

[108] Sen, Z.: Solar energy fundamentals and modeling techniques: atmosphere, environment, climate change and renewable energy. Springer Science & Business Media (2008)

[109] Shahid, F., Zameer, A., Muneeb, M.: A novel genetic lstm model for wind power forecast. Energy 223, 120069 (2021)

[110] Shao, B., Song, D., Bian, G., Zhao, Y.: Wind speed forecast based on the lstm neural network optimized by the firework algorithm. Advances in Materials Science and Engineering 2021, 1–13 (2021)

[111] Shapiro, S.S., Francia, R.: An approximate analysis of variance test for normality. Journal of the American statistical Association 67(337), 215–216 (1972)

[112] Shi, J., Guo, J., Zheng, S.: Evaluation of hybrid forecasting approaches for wind speed and power generation time series. Renewable and Sustainable Energy Reviews 16(5), 3471–3480 (2012)

[113] Silvestrini, S., Lavagna, M.: Deep learning and artificial neural networks for spacecraft dynamics, navigation and control. Drones 6(10), 270 (2022)

[114] Simon, D.: Evolutionary optimization algorithms. John Wiley & Sons (2013)

[115] Stankovic, M., Antonijevic, M., Bacanin, N., Zivkovic, M., Tanaskovic, M., Jovanovic, D.: Feature selection by hybrid artificial bee colony algorithm for intrusion detection. In: 2022 International Conference on Edge Computing and Applications (ICECAA). pp. 500–505. IEEE (2022)

[116] Stegherr, H., Heider, M., Hähner, J.: Classifying metaheuristics: Towards a unified multi-level classification system. Natural Computing pp. 1–17 (2020)

[117] Stoean, C., Zivkovic, M., Bozovic, A., Bacanin, N., Strulak-Wójcikiewicz, R., Antonijevic, M., Stoean, R.: Metaheuristic-based hyperparameter tuning for recurrent deep learning: Application to the prediction of solar energy generation. Axioms 12(3), 266 (2023)

[118] Strumberger, I., Tuba, E., Zivkovic, M., Bacanin, N., Beko, M., Tuba, M.: Dynamic search tree growth algorithm for global optimization. In: Doctoral Conference on Computing, Electrical and Industrial Systems. pp. 143–153. Springer (2019)

[119] Talbi, E.G.: Metaheuristics: from design to implementation. John Wiley & Sons (2009)

[120] Tayebi, M., El Kafhali, S.: Performance analysis of metaheuristics based hyperparameters optimization for fraud transactions detection. Evolutionary Intelligence pp. 1–19 (2022)

[121] Twidell, J., Weir, T.: Renewable energy resources, 2006. Taylor and Francis, ISBN 0–419–25330–0. Schirmer, P., et al., Experimental investigation of the performance of the solar tunnel dryer for drying bananas. Journal of Renewable Energy 2, 119–129 (1996)

[122] Wang, D., Cui, X., Niu, D.: Wind power forecasting based on lstm improved by emd-pca-rf. Sustainability 14(12), 7307 (2022)

[123] Wang, D., Cui, X., Niu, D.: Wind power forecasting based on lstm improved by emd-pca-rf. Sustainability 14(12) (2022)

[124] Wang, H., Zhang, Y., Liang, J., Liu, L.: Dafa-bilstm: Deep autoregression feature augmented bidirectional lstm network for time series prediction. Neural Networks 157, 240–256 (2023)

[125] Wang, L., Liu, H., Pan, Z., Fan, D., Zhou, C., Wang, Z.: Long short-term memory neural network with transfer learning and ensemble learning for remaining useful life prediction. Sensors 22(15), 5744 (2022)

[126] Wang, N., Li, Z.: Short term power load forecasting based on bes-vmd and cnn-bi-lstm method with error correction. Frontiers in Energy Research 10 (2023)

[127] Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE transactions on evolutionary computation 1(1), 67–82 (1997)

[128] Wright, S.J.: Continuous optimization (nonlinear and linear programming). Foundations of Computer-Aided Process Design 7 (1999)

[129] Xiong, J., Peng, T., Tao, Z., Zhang, C., Song, S., Nazir, M.S.: A dual-scale deep learning model based on elm-bilstm and improved reptile search algorithm for wind power prediction. Energy p. 126419 (2022)

[130] Yang, X.S.: Firefly algorithms for multimodal optimization. In: International symposium on stochastic algorithms. pp. 169–178. Springer (2009)

[131] Yang, X.S.: Nature-Inspired Metaheuristic Algorithms (07 2010)

[132] Yang, X.S.: A new metaheuristic bat-inspired algorithm. In: Nature inspired cooperative strategies for optimization (NICSO 2010), pp. 65–74. Springer (2010)

[133] Yang, X.S., Gandomi, A.H.: Bat algorithm: a novel approach for global engineering optimization. Engineering computations (2012)

[134] Zhang, T., Tang, Z., Wu, J., Du, X., Chen, K.: Multi-step-ahead crude oil price forecasting based on two-layer decomposition technique and extreme learning

machine optimized by the particle swarm optimization algorithm. Energy 229, 120797 (2021)

[135] Zhang, Y., Liu, K., Qin, L., An, X.: Deterministic and probabilistic interval prediction for short-term wind power generation based on variational mode decomposition and machine learning methods. Energy Conversion and Management 112, 208–219 (2016)

[136] Zhou, D., Zhuang, X., Zuo, H., Cai, J., Zhao, X., Xiang, J.: A model fusion strategy for identifying aircraft risk using cnn and att-bilstm. Reliability Engineering & System Safety 228, 108750 (2022)

[137] Zivkovic, M., Bacanin, N., Antonijevic, M., Nikolic, B., Kvascev, G., Marjanovic, M., Savanovic, N.: Hybrid cnn and xgboost model tuned by modified arithmetic optimization algorithm for covid-19 early diagnostics from x-ray images. Electronics 11(22), 3798 (2022)

[138] Zivkovic, M., Bacanin, N., Tuba, E., Strumberger, I., Bezdan, T., Tuba, M.: Wireless sensor networks life time optimization based on the improved firefly algorithm. In: 2020 International Wireless Communications and Mobile Computing (IWCMC). pp. 1176–1181. IEEE (2020)

[139] Zivkovic, M., Bacanin, N., Venkatachalam, K., Nayyar, A., Djordjevic, A., Strumberger, I., Al-Turjman, F.: Covid-19 cases prediction by using hybrid machine learning and beetle antennae search approach. Sustainable Cities and Society 66, 102669 (2021)

[140] Zivkovic, M., Bezdan, T., Strumberger, I., Bacanin, N., Venkatachalam, K.: Improved harris hawks optimization algorithm for workflow scheduling challenge in cloud–edge environment. In: Computer Networks, Big Data and IoT, pp. 87–102. Springer (2021)

[141] Zivkovic, M., Venkatachalam, K., Bacanin, N., Djordjevic, A., Antonijevic, M., Strumberger, I., Rashid, T.A.: Hybrid genetic algorithm and machine learning method for covid-19 cases prediction. In: Proceedings of International Conference on Sustainable Expert Systems: ICSES 2020. vol. 176, p. 169. Springer Nature (2021)

[142] Zivkovic, M., Zivkovic, T., Venkatachalam, K., Bacanin, N.: Enhanced dragonfly algorithm adapted for wireless sensor network lifetime optimization. In: Data Intelligence and Cognitive Informatics, pp. 803–817. Springer (2021)

[143] Züttel, A., Gallandat, N., Dyson, P.J., Schlapbach, L., Gilgen, P.W., Orimo, S.I.: Future swiss energy economy: the challenge of storing renewable energy. Frontiers in Energy Research 9, 785908 (2022)