УНАПРЕЂЕЊЕ АЛГОРИТАМА РОЈЕВА ЧЕСТИЦА ЗА ПРОБЛЕМЕ КЛАСИФИКАЦИЈЕ И РЕГРЕСИЈЕ СА ПРИМЕРИМА ПРИМЕНЕ

By Aleksandar Petrović

WORD COUNT 38067 TIME SUBMITTED 05-JUN-2025 12:20PM

PAPER ID 116603077



УНИВЕРЗИТЕТ СИНГИДУНУМ

ДЕПАРТМАН ЗА ПОСЛЕДИПЛОМСКЕ СТУДИЈЕ

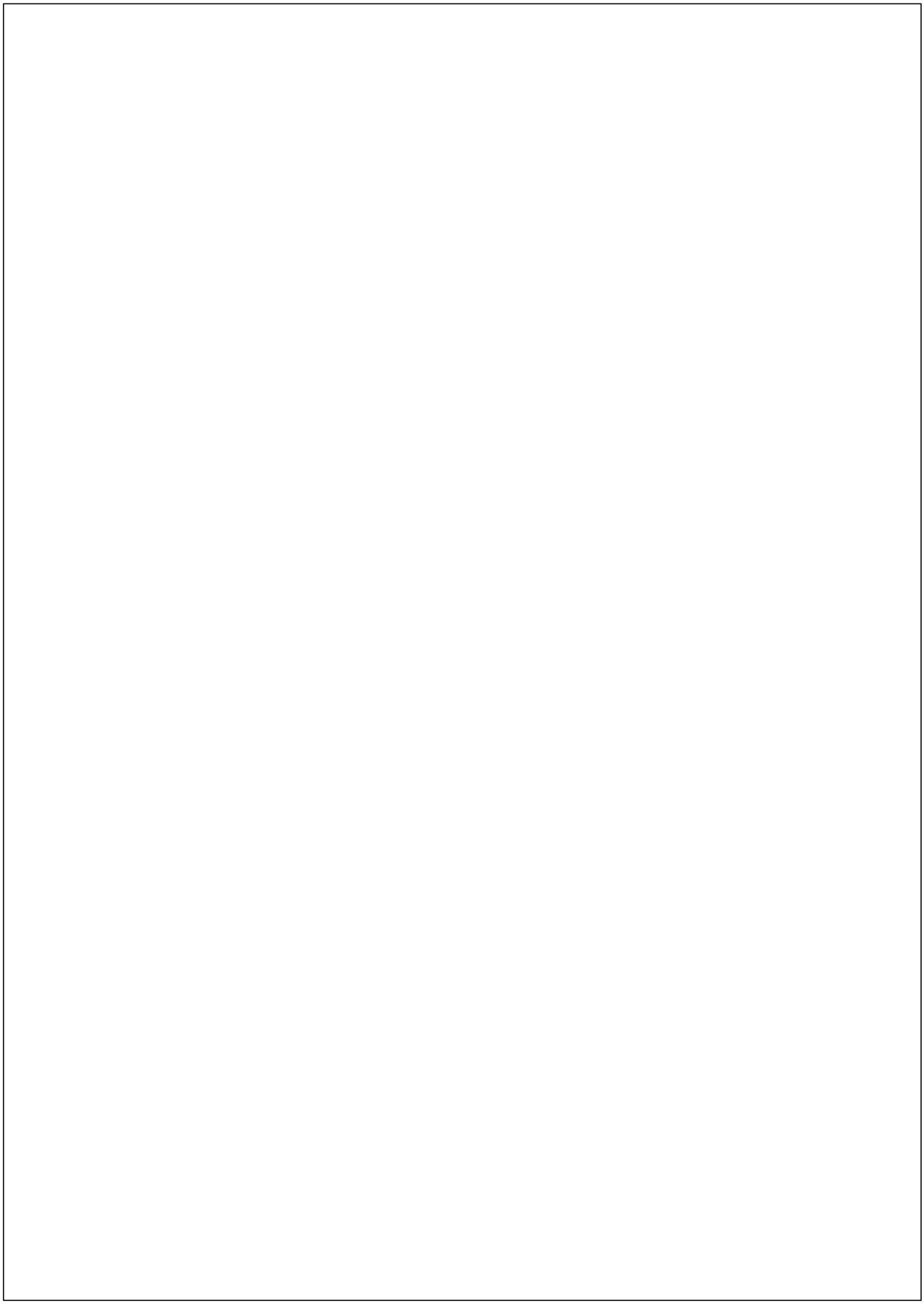
ДОКТОРСКА ДИСЕРТАЦИЈА

УНАПРЕЂЕЊЕ АЛГОРИТАМА РОЈЕВА ЧЕСТИЦА ЗА ПРОБЛЕМЕ КЛАСИФИКАЦИЈЕ И РЕГРЕСИЈЕ СА ПРИМЕРИМА ПРИМЕНЕ

МЕНТОР:
проф. др Милош Станковић

СТУДЕНТ: Александар Петровић
БРОЈ ИНДЕКСА: 480158/2022

БЕОГРАД, 2025. ГОД.





SINGIDUNUM UNIVERSITY
DEPARTMENT OF POSTGRADUATE STUDIES

DOCTORAL DISSERTATION

IMPROVEMENT OF PARTICLE SWARM ALGORITHMS FOR CLASSIFICATION AND REGRESSION PROBLEMS WITH APPLICATION EXAMPLES

MENTOR:
prof. Miloš Stanković, PhD

STUDENT: Aleksandar Petrović
INDEX NUMBER: 480158/2022

BELGRADE, 2025.

Наслов:	Унапређење алгоритама ројева честица за проблеме класификације и регресије са примерима примене
Аутор:	Александар Петровић
Ментор:	проф. др Милош Станковић
Комисија:	проф. др Милош Станковић проф. др Миодраг Живковић проф. др Бошко Николић
Кључне речи:	оптимизација метахеуристике интелигенција ројева оптимизација ројевима честица хибридизација

Copyright © 2025 Александар Петровић
Универзитет Сингидунум

Title:	Improvement of particle swarm algorithms for classification and regression problems with application examples
Author:	Aleksandar Petrović
Mentor:	prof. Miodrag Živković, PhD
Commission:	prof. Miloš Stanković, PhD prof. Miodrag Živković, PhD prof. Boško Nikolić, PhD
Keywords:	optimization metaheuristics swarm intelligence particle swarm optimization hybridization

Copyright © 2025 Aleksandar Petrović
Singidunum University

Захвалница

За Сању и Живорада.

Ово истраживање је подржао Фонд за науку Републике Србије бр. гранта 7502, *Intelligent Multi-Agent Control and Optimization Applied to Green Buildings and Environmental Monitoring Drone Swarms (ECOSwarm)*.

Београд, 2025. год. Александар Петровић

Acknowledgments

For Sanja and Živorad.

This research was supported by the Science Fund of the Republic of Serbia, and grant No. 7502, Intelligent Multi-Agent Control and Optimization Applied to Green Buildings and Environmental Monitoring Drone Swarms (ECOSwarm).

Belgrade, 2025. Aleksandar Petrović

Сажетак

Представљени рад је настао као истраживање алгоритама интелигенције ројева (енгл. - *swarm intelligence - SI*) и експеримената који их примењују за решавање проблема из стварног света. Главни циљ овог истраживања јесте проналажење побољшаних верзија алгоритма оптимизације ројевима честица (енгл. - *particle swarm optimization - PSO*) који ефикасније решавају проблеме из стварног живота. Примењени су различити принципи хибридизације са другим метахеуристикама и стратегијама иницијализације са циљем проналажења бољег баланса фаза експлорације и експлоатације приликом решавања проблема појединачно. Поред овог алгоритма примењени су и различити метахеуристички оптимизатори како би се перформансе предложених решења за два различита проблема стварног живота могле потврдити. Прва примена предложене методологије се односи на проблеме поморског саобраћаја, односно идентификацију и праћење пловила. Друга примена се бави другачијим али не мање важним проблемом предикције грешке у програмском коду. За оба случаја је поред компаративне анализе са постојећим алгоритмима високих перформанси примењена статистичка валидација резултата и анализа. Резултат истраживања су две нове верзије *PSO* које су представљене и објављене у престижним научним часописима.

Abstract

The presented work was oriented as a research on swarm intelligence (SI) algorithms and experiments that apply them to solve real-world problems. The main goal of this research is to find improved versions of the particle swarm optimization algorithm (PSO) that solve real-world problems more efficiently. Different hybridization principles with other metaheuristics and initialization strategies were applied with the aim of finding a better balance between the exploration and exploitation phases when solving problems individually. In addition to this algorithm, different metaheuristic optimizers were also applied in order to validate the performance of the proposed solutions for two different real-world problems. The first application of the proposed methodology relates to maritime traffic problems, namely vessel identification and tracking. The second application deals with a different but not less important problem of error prediction in program code. In both cases, in addition to comparative analysis with existing high-performance algorithms, statistical validation of the results and analysis was applied. The results of the research are two new versions of PSO which were both presented and published in prestigious scientific journals.

Садржај

1 Увод **2**
- 1.1 Циљеви рада и значај истраживања . . . 4
- 1.2 Полазне хипотезе . . . 5
- 1.3 Методе истраживања . . . 6
- 1.4 Научни доприноси . . . 6
- 1.5 Структура дисертације . . . 7

2 Опис проблема **9**
- 2.1 Експерименти предвиђања поморског саобраћаја . . . 9
- 2.2 Експерименти препознавања грешке у програмском коду . . . 11

3 Метахеуристичка оптимизација **14**
- 3.1 Оптимизација роjевима честица . . . 15
- 3.2 Генетски алгоритам . . . 17
- 3.3 Оптимизација колонијама пчела . . . 18
- 3.4 Алгоритам слепог миша . . . 19
- 3.5 Алгоритам оптимизације китова . . . 20
  - 3.5.1 Окруживање плена . . . 20
  - 3.5.2 Напад методом мреже мехурића . . . 21
  - 3.5.3 Потрага за пленом . . . 21
- 3.6 Оптимизација Харисових јастребова . . . 23
- 3.7 Алгоритам оптимизације шимпанзи . . . 25
- 3.8 Претрага променљивог суседства . . . 28
- 3.9 Оптимизација хиперболичким синусом и косинусом . . . 28
- 3.10 *COLSHADE* . . . 33

4 Остале методе **37**
- 4.1 Обрада природног језика . . . 37
  - 4.1.1 Двосмерни енкодер из репрезентација трансформатора . . . 38
  - 4.1.2 Техника учесталости појма - инверзне учесталости документа . . . 38
- 4.2 Конволуцијске неуронске мреже . . . 39
- 4.3 Дуга краткорочна меморија . . . 39
- 4.4 Адаптивни бустинг . . . 41
- 4.5 Екстремни градијентни бустинг . . . 41

5 Модификације алгоритама **43**
- 5.1 Модификације алгоритма роjева честица према проблемима поморског саобраћаја . . . 43

5.1.1 Бустована оптимизација роjевима честица . . . 43
5.1.2 Псеудокод модификованог алгоритма . . . 45
5.2 Модификације алгоритма оптимизације роjевима честица према проблемима детекције грешака у коду . . . 45
5.2.1 Мулти-популацијска оптимизација роjевима честица . . . 46
5.2.2 Комплексност предложене мулти-популацијске оптимизације роjевима честица . . . 48

6 Резултати **49**
6.1 Примењене метрике за компаративну анализу . . . 49
6.2 Експерименти предвиђања поморског саобраћаја . . . 50
6.2.1 Примењени скупови података и предобрада података . . . 51
6.2.2 Подешавања експеримената . . . 52
6.2.3 Резултати, компаративна анализа, валидација и визуелизација . . . 53
6.2.4 Статистичка евалуација . . . 62
6.2.5 Интерпретација резултата најбољих модела . . . 63
6.3 Експерименти препознавања грешке у програмском коду . . . 66
6.3.1 Поставка експеримената . . . 66
6.3.2 Симулације *CNN* класификације на првом слоју . . . 70
6.3.3 Симулације *AdaBoost* класификације на другом слоју . . . 77
6.3.4 Симулације *XGBoost* класификације на другом слоју . . . 83
6.3.5 Симулације *AdaBoost* класификације са *NLP* метода на другом слоју . . . 89
6.3.6 Симулације *XGBoost* класификације са *NLP* методама на другом слоју . . . 95
6.3.7 Анализа . . . 101
6.3.8 Статистичка валидација . . . 102
6.3.9 Интерпретације најбољег модела . . . 103

7 Закључна разматрања **106**

Списак слика

2.1 Дијаграм тока предложеног радног оквира за експерименте предвиђања поморског саобраћаја. 11
2.2 Дијаграм тока предложеног радног оквира за експерименте грешке у програмском коду. 13

5.1 Механизми укрштања и мутације. 47

6.1 Бар дијаграм дистрибуције класа. 51
6.2 Кружни дијаграм дистрибуције класа. 52
6.3 Дијаграми класификације пловила - минимизација грешке. 56
6.4 Дијаграми класификације пловила - максимизација Коеновог Капа параметра. 59
6.5 Дијаграми предвиђања путање пловила. 62
6.6 Сумарна важност обележја одређена путем *XGBoost* и *SHAP*. 64
6.7 Важност обележја одређена путем *XGBoost* и *SHAP*. 65
6.8 Графикони важности обележја за теретна пловила. 65
6.9 Графикони важности обележја за тегљаче. 66
6.10 Дистрибуција класа у HACA скупу података. 67
6.11 Корелација обележја у HACA скупу података. 68
6.12 Дистрибуције циљне и индикаторске функције *L1 CNN* модела. 71
6.13 Конвергенција циљне функције *L1 CNN* модела. 72
6.14 Конвергенција индикаторске функције *L1 CNN* модела. 73
6.15 *ROC* крива *L1 CNN-MPPSO* оптимизованог модела. 75
6.16 Матрица конфузије *L1 CNN-MPPSO* оптимизованог модела. 75
6.17 Визуелизација *CNN-MPPSO* оптимизованог *L1* модела. 76
6.18 Конвергенција циљне функције *L2 Adaboost* модела. 78
6.19 Дистрибуције циљне и индикаторске функције *L2 AdaBoost* модела. . . 79
6.20 Конвергенција индикаторске функције *L2 Adaboost* модела. 80
6.21 *ROC* крива *L2 AdaBoost* оптимизованог модела. 82
6.22 Матрица конфузије *L2 AdaBoost* оптимизованог *L1* модела. 82
6.23 Дистрибуције циљне и индикаторске функције *L2 XGBoost* модела. . . 84
6.24 Конвергенција циљне функције *L2 XGBoost* модела. 85
6.25 Конвергенција индикаторске функције *L2 XGBoost* модела. 86
6.26 *ROC* крива *L2 XGBoost* оптимизованог *L1* модела. 88
6.27 Матрица конфузије *L2 XGBoost* оптимизованог *L1* модела. 88
6.28 Дистрибуција циљне и индикаторске функције *NLP L2 Adaboost* модела. 90
6.29 Конвергенција циљне функције *NLP L2 AdaBoost* модела. 91
6.30 Конвергенција индикаторске функције *NLP L2 AdaBoost* модела. . . . 92
6.31 *ROC* крива *NLP L2 AdaBoost* оптимизованог *L1* модела. 94

6.32 Матрица конфузије *NLP L2 AdaBoost* оптимизованог *L1* модела. 94
6.33 Дистрибуција циљне и индикаторске функције *NLP L2 XGBoost* модела. 96
6.34 Конвергенција циљне функције *NLP L2 XGBoost* модела. 97
6.35 Конвергенција индикаторске функције *NLP L2 XGBoost* модела. 98
6.36 *ROC* крива *NLP L2 XGBoost* оптимизованог *L1* модела. 100
6.37 Матрица конфузије *NLP L2 XGBoost* оптимизованог *L1* модела. 100
6.38 *KDE* дијаграми пет експеримената предвиђања софтверске грешке. . . 102
6.39 Важност обележја за најбоље моделе *CNN*, *XGBoost* и *AdaBoost* на НАСА скупу података. 104
6.40 Важност обележја за најбоље моделе *AdaBoost* на *NLP* скупу података. 104
6.41 Важност обележја за најбоље моделе *XGBoost* на *NLP* скупу података. 105

Списак табела

6.1 Укупни резултати циљне функције (грешка класификације) - Експеримент минимизације грешке класификације. 54
6.2 Укупни резултати индикаторске функције (Коенова Капа) - Експеримент минимизације грешке класификације. 54
6.3 Детаљне метрике по класама за најбољи модел *XG-BPSO* метахеуристике - Експеримент минимизације грешке класификације. 55
6.4 Најбољи селектовани хиперпараметри вишекласне класификације - Експерименти минимизације грешке класификације. 55
6.5 Укупни резултати циљне функције (Коенова Капа) - Експерименти максимизације Коеновог Капа параметра. 57
6.6 Укупни резултати индикаторске функције (грешка класификације) - Експерименти максимизације Коеновог Капа параметра. 57
6.7 Детаљне метрике по класама за најбољи модел *XG-BPSO* метахеуристике - Експерименти максимизације Коеновог Капа параметра. 58
6.8 Најбољи селектовани хиперпараметри вишекласне класификације - Експерименти максимизације Коеновог Капа параметра. 58
6.9 Укупни резултати циљне функције - Експерименти предвиђања путање пловила. 60
6.10 Детаљне метрике за најбољи модел - Експерименти предвиђања путање пловила. 60
6.11 Најбољи селектовани хиперпараметри - Експерименти предвиђања путање пловила. 61
6.12 Шапиро-Вилк тестови нормалности. 63
6.13 Резултати Вилкоксоновог теста упарених узорака који показују p-вредности за сва три експеримента (*BPSO* у односу на остале). 63
6.14 Опсези *CNN* параметара за оптимизацију. 69
6.15 Опсези *AdaBoost* параметара за оптимизацију. 69
6.16 Опсези *XGBoost* параметара за оптимизацију. 69
6.17 Исходи циљне функције *L1 CNN* модела. 70
6.18 Исходи индикаторске функције *L1 CNN* модела. 70
6.19 Детаљно поређење модела са најбољим учинком *L1 CNN* симулација. . 74
6.20 Селектовани параметри оптимизованих *CNN* модела. 76
6.21 Исходи циљне функције *L2 AdaBoost* модела. 77
6.22 Исходи индикаторске функције *L2 AdaBoost* модела. 77
6.23 Детаљно поређење метрика модела са најбољим учинком *L2 AdaBoost* симулација. 81
6.24 Селектовани параметри најбоље оптимизованих *L2 AdaBoost* модела. . 83
6.25 Исходи циљне функције *L2 XGBoost* модела. 83

6.26 Исходи индикаторске функције *L2 XGBoost* модела. 83
6.27 Детално поређење метрика модела са најбољим учинком *L2 XGBoost* симулација. 87
6.28 Селектовани параметри најбоље оптимизованих *L2 XGBoost* модела. . 89
6.29 Исходи циљне функције *NLP L2 AdaBoost* модела. 89
6.30 Исходи индикаторске функције *NLP L2 AdaBoost* модела. 89
6.31 Детално поређење метрика модела са најбољим учинком *NLP L2 AdaBoost* симулација. 93
6.32 Селектовани параметри најбоље оптимизованих *NLP L2 AdaBoost* модела. 95
6.33 Исходи циљне функције *NLP L2 XGBoost* модела. 95
6.34 Исходи индикаторске функције *NLP L2 XGBoost* модела. 96
6.35 Детално поређење метрика модела са најбољим учинком *NLP L2 XGBoost* симулација. 99
6.36 Селектовани параметри најбоље оптимизованих *NLP L2 XGBoost* модела. 101
6.37 Поређење са резултатима основних модела за HACA експеримент. . . . 101
6.38 Поређење са резултатима основних модела за *NLP* експеримент. 101
6.39 Резултати пет спроведених Шапиро-Вилко симулација за верификацију услова нормалности за безбедно коришћење параметарских тестова. 103
6.40 Резултати Вилкоксоновог теста упарених узорака за пет спроведених симулација. 103

Списак алгоритама

1 Псеудокод *PSO*. 17
2 Псеудокод *GA*. 17
3 Псеудокод *ABC*. 18
4 Псеудокод *BA*. 20
5 Псеудокод *WOA*. 22
6 Псеудокод *HHO*. 25
7 Псеудокод *ChOA*. 27
8 Псеудокод *VNS*. 28
9 Псеудокод *SCHO*. 32
10 Псеудокод *levy/1/bin mutation*. 35
11 Псеудокод *COLSHADE*. 36
12 Псеудокод *BPSO* алгоритма. 45
13 Псеудокод *MPPSO*. 48

Листа скраћеница

ГПС - глобални позициони систем

ДНК - дезоксирибонуклеинска киселина

НАСА - Национална агенција за аеронаутику и свемир

НП - недетерминистичка полиномијална комплексност

П - детерминистичко полиномско време

САД - Сједињене Америчке Државе

ABC - оптимизација колонијама пчела

AdaBoost - адаптивни бустинг

AGI - општа вештачка интелигенција

AIS - систем за аутоматску идентификацију

AOA - аритметичка оптимизација

BA - алгоритам слепог миша

BERT - двосмерни енкодер из репрезентација трансформатора

BPSO - бустована оптимизација ројевима честица

CEL - елитна хаотична стратегија обучавања

ChOA - алгоритам оптимизације шимпанзи

CNN - конволуционе неуронске мреже

CR - стопа укрштања

DBSCAN - просторно груписање за примене са шумом

DE/cpb - мутациони оператор диференцијалне еволуције/тренутне до *pbest* мутације

DL - дубоко учење

EDE - грешка еуклидског растојања

FFE - евалуација функције погодности

GA - генетски алгоритам

HHO - оптимизације Харисових јастребова

HSC - брзо пловило

IoA - индекс усклађености

IoT - интернет ствари

KDE - процена густине језгра

KLOC - број дефеката на хиљаду линија кода

LSTM - дуга краткорочна меморија

L1 - први слој

L2 - други слој

MAE - средња апсолутна грешка

MCC - Метјузов коефицијент корелације

MLM - модел језичке маске

MLP - вишеслојни перцептрон

MPPSO - мулти-популацијска оптимизација роjевима честица

MSE - средња квадратна грешка

NFL - теорема бесплатног ручка

NLP - обрада природног језика

NN - неуронска мрежа

PSO - оптимизација роjевима честица

RMSE - корен средње квадратне грешке

RNN - рекурентна неуронска мрежа

R^2 - коефицијент одлучности

SAR - потрага и спашавање

SCA - синус косинус алгоритам

SCHO - оптимизација хиперболичким синусом и косинусом

SDLC - животни циклус развоја софтвера

SDP - предикција дефекта софтвера

SF - фактор скалирања

SI - интелигенција роjева

std - стандардна девијација

TF-IDF - техника учесталости појма - инверзне учесталости документа

var - варијанса

VNS - претрага промењивог суседства

QRL - квази-рефлексивно учење

WOA - алгоритам оптимизације китова

XGBoost - екстремно појачавање градијента

Глава 1

Увод

Истраживање представљено у овом раду се односи на побољшање *PSO* алгоритма за различите примене у стварном животу. Истраживање представља две унапређене верзије *PSO* у зависности од типа проблема за који их је потребно применити. Овај алгоритам спада у групу метахеуристичких решења из области вештачке интелигенције. Вештачка интелигенција обухвата више подобласти од којих је тренутно највећи успех постигло машинско учење (енгл. - *machine learning*), дубоко учење (енгл. - *deep learning*), неуронске мреже (енгл. - *neural networks - NN*), обрада природног језика (енгл. - *natural language processing - NLP*) и рачунарски вид (енгл. - *computer vision*) [1]. Главна одлика техничких решења из ове области је способност учења из датих примера. Напредак ове области је омогућен повећањем доступности података, али и приступачност рачунарских ресурса високих перформанси по моделу плаћања по потрошњи уз рачунарство у облаку (енгл. - *cloud computing*). Предност решења из ове области је и то што могу имати најразличитије примене.

Једна од основних теорема на којој се заснива представљено истраживање се зове теорема бесплатног ручка (енгл. - *no free lunch theorem - NFL*) [2]. По овој теореми не постоји универзални алгоритам који може решити све проблеме. Не постоји ни алгоритам који може решити све случајеве једног проблема. На пример ако имамо два скупа података која су наизглед везана за сличну тему, перформансе нису загарантоване на другом скупу уколико је модел треniран на првом. Ово је управо и разлог из којег је неопходно да сваки случај примене алгоритама базираних на вештачкој интелигенцији буде прилагођен посебно за проблем који решава. Из ових разлога је неопходна и од пресудног значаја оптимизација, односно процес прилагођавања и подешавања метода вештачке интелигенције за дати проблем. Ова подешавања у области машинског учења се називају хиперпараметри. Другим речима, метода покушаја и грешке (енгл. - *trial and error*) није применљива и потребно је пронаћи ефикаснији начин за проналажење решења.

Овај процес се односи управо на хиперпараметре модела и за циљ има да пронађе њихове оптималне вредности. Појам оптималности је условљен применом за коју се модел подешава, али се може дефинисати као најбољи скуп вредности. То значи да сам процес оптимизације хиперпараметара не тражи само најбољу појединачну вредност одређеног параметра, већ најбоље вредности, односно комбинацију свих хиперпараметара тог модела по одређеном критеријуму. Комплексност оваквих израчунавања може бити висока, што значи да ни овај приступ проблему није без недостатака.

Употребом машинског учења се решава проблем предикције, а неке од важнијих

примена су из области медицине, агрикултуре, опште и рачунарске безбедности. Напреци у *NLP* обла113постигнути *ChatGPT* и сличним конверзацијским агентима су приближили циљ опште вештачке интелигенције (енгл. - *artificial general intelligence - AGI*). Сматра се да је *GhatGPT* успешно прошао Турингов тест који је до скоро сматран најбољом методом тестирања квалитета вештачке интелигенције. Нажалост, успешан исход овог теста није довео до *AGI* већ до тога да су неопходне нове методе за утврђивање нивоа софистицираности вештачке интелигенције.

Поред широке примене у стварном свету, применом апроксимативних метода се успешно решавају задаци као што је оптимизација хиперпараметара. Важно је напоменути да и поред високе комплексности неопходних израчунавања, овим методама је и даље далеко ефикасније проналажење оптималног скупа хиперпараметара од *trial and error* методе. Проналажење оптималног подскупа хиперпараметара уопштено говорећи није решиво у полиномијалном времену, те се за овакав тип проблема каже да имају недетерминистичку полиномијалну комплексност (НП), односно да је проблем НП тежак.

Значај за област оптимизације је остварила подобласт вештачке интелигенције позната као *SI* [3]. Кључна карактеристика која ову групу чини посебном јесте да су алгоритми који јој припадају засновани на принципима заснованим на концепту ројева, односно организације великих група јединки у природи. Идеја овог приступа јесте да се за тешке оптимизационе проблеме који нису решиви традиционалним методама, пронађе решење применом принципа када јединка није способна сама да изврши задатак. Ова област је препознатљива по решењима која су погодна за оптимизовање других метода јер пружа боље могућности проналаска оптималног решења у поређењу са класичним методама оптимизације. Решења из ове групе алгоритама се такође називају метахеуристичким. Претпоставља се да математички модели који су инспирисани неком појавом из природе или низом догађаја, понашања група јединки и слично, добро решавају тешке оптимизационе проблеме јер праћење шаблона које природа оставља за собом врло често указује на ефикаснији пут него што је човек пронашао. Тако и код алгоритама ројева, пратећи правила на пример мрављe колоније, могуће је решити комплексне проблеме из стварног живота. Међутим, ови алгоритми нису без мана. За њих су карактеристичне две фазе, експлорација и експлоатација, у којима се понашање алгоритма мења из опширне претраге у фокусирано истраживање мањег подручја. Пошто принципи претраге имају фундаменталне разлике у ове две фазе може се закључити да је код већине алгоритама једна од ове две фазе доминантна.

Комбиновање више метода из области вештачке интелигенције доводи до новог приступа проблемима који се зове хибридизација [4]. Узимајући у обзир *NFL* теорему, јасно је да су одређени алгоритми бољи у општој претрази, а неки други у уској претрази [5]. Хибридизација спаја два алгоритама како би баланс између ове две фазе био постигнут [6]. Овај концепт је изродио мноштво алгоритама и њихов број ће само расти. Ову тврдњу поткрепљује и чињеница да је примена хибридних метахеуристика готово неограничена, јер их је могуће применити на све што се може представити скупом података. Циљ хибридизације је унапређене перформанси, што се може оценити путем просека резултата алгоритама на стандардизованим тестовима.

Сумирано, *SI* група алгоритама се показала као одлична за оптимизацију алгоритама машинског учења. Увођењем метода као што су хибридизација, број могућих решења из ове групе алгоритама се драстично повећава. У складу са *NFL* теоремом,

са повећањем броја решења, њена антитеза да један алгоритам решава све проблеме је теже остварива. Након дужег објашњења како је истраживање вршено, могуће је представити садржај, односно тему овог рада читаоцу. У овом раду, основни *PSO* је модификован на два начина и тестиран у две групе експеримената. Алгоритам из *SI* групе је тестиран у две групе експеримената. Различите модификације и укрштања са другим алгоритмима су неопходна јер се примена алгоритма знатно разликује у овим експериментима. Поред две главне групе, извршено је доста додатних тестова различитих комбинација осталих решења у радним оквирима представљеним за описане проблеме. Ови тестови укључују тестове са индикаторским функцијама, поред основних тестова циљне функције, као и статистичку валидацију постигнутих резултата.

Прва група проблема на којима је примењена предложена методологија се односи на предикције везане за поморски саобраћај. Вода је још увек један део природе који човек није у потпуности освојио и сурови услови који одликују свако овакво путовање само указују на њихову проблематику. Системи који се тренутно користе за лоцирање брода имају мане, чему сведочи велики број злоупотреба као што су технике риболова које су опасне по животну средину као и друге илегалне радње због којих би починиоци желели да сакрију траг. Без драстичних измена по питању опреме која се користи за праћење и идентификацију пловила, могуће је постићи знатна унапређења ових система применом вештачке интелигенције. Напредак у овој области је омогућио да се на основу познатих података са високом прецизношћу предвиди потенцијална позиција пловила, као и величина и карактеристике пловила.

Другу групу експеримената чине симулације проналажења грешке у програмском коду. Ова примена се знатно разликује од претходног проблема и тако исто се разликује и предложена методологија за унапређење овог алгоритма. Потенцијал вештачке интелигенције у проналажењу грешака је препознат од почетка масовне употребе генеративних конверзацијских агената, а корак даље јесте предвиђање дела кода на којем ће се појавити грешка. Управо је за ову потребу модификована основна верзија поменутог алгоритама. Поред бројних тестова који потврђују перформансе унапређеног модела, сви примењени алгоритми су тестирани са и без примене *NLP* метода. Закључак ових тестова је био да *NLP* може знатно побољшати резултате када је у питању проблем грешке у коду, што је и логично ако се узме у обзир да је програмски код такође врста језика.

1.1 Циљеви рада и значај истраживања

У овој докторској тези су предложена два нова алгоритама, а главна два научна рада која је чине се баве оптимизацијом *PSO* алгоритма. Предложене методологије су настале применом хибридизације, али и увођењем нових метода претраге. Ово би се могло сматрати за главни циљ тезе, али поред тога је и значајна из вишеструких разлога. Поред оптимизације постојећег алгоритма, предложена решења су примењена на проблеме из стварног живота. За оба експеримента су предложене промене на основни алгоритам ригорозно тестиране. Постојећи алгоритми из литературе који показују високе перформансе за теме обрађене у овом раду су такође тестирани под истим условима као и предложено решење зарад валидације резултата.

Описани проблеми поморског саобраћаја и детекције грешака у коду имају висок значај за стварни свет. Да би се постигла унапређења при решавању ових проблема

потребна је примена вештачке интелигенције. Циљ представљеног истраживања је да докаже могућност примене вештачке интелигенције за наведене проблеме, како у пракси, тако и у теорији. Извршене анализе алгоритама и резултата служе да прикажу предности и мане тестираних модела. Потребне промене традиционалних система за решавање проблема из стварног света истражених у овом раду треба да побољшају сигурност поморског путовања, заштите животну средину, али и квалитет живота програмера у случају проналаска грешке у коду.

Општи циљ тезе је проналазак нових, унапређених алгоритама на бази *PSO*, њихова примена у оптимизацији хиперпараметара одређених модела машинског учења као и њихова верификација на проблемима унапређења пловидбе и израде софтвера.

Практични и технички циљ се може описати развојем и применом различитих радних оквира у зависности од потребе за решавање проблема пловидбе и грешака у софтверу. Могућности за решавање ових проблема су анализиране са акцентом на безбедност.

Друштвени циљ тезе се може огледати у подржавању глобалних логистичких система и њихове безбедности.

Узимајући у обзир наведене погледе и циљеве, очекивани резултати истраживања су решавање тренутних проблема у поморском саобраћају и развоју софтвера и унапређења алгоритама. Имплементацијом напредних система навођења пловила и њихове класификације могућа су унапређења у виду безбедније и потенцијално брже путање, што би и смањило трошкове. Применом вештачке интелигенције на проблем предикције дефекта софтвера (енгл. - *software defect prediction - SDP*) сличне промене би требало да су приметне.

Истраживање је извршено на јавним скуповима података а неки од њих су прикупљани од стране државних установа у Сједињеним Америчким Државама (САД). Резултати осликавају перформансе модела, комплексност, као и утицај обележја из скупа података над којим је експериментисано.

На описани начин су примењене методе анализе и синтезе за потребе доказивања или оповргавања постављених хипотеза. Истраживање је верификовано кроз бројне објављене научне радове, као и одобравање стручне и научне заједнице.

1.2 Полазне хипотезе

Општа хипотеза: унапређење *PSO* алгоритма за оптимизацију хиперпарематара у проблемима пловидбе и израде софтвера.

Остале хипотезе:

- **Хипотеза (Х1)**: Предложени радни оквири базирани на више различитих врста решења из области вештачке интелигенције се могу применити у стварном животу са високом прецизношћу.
- **Хипотеза (Х2)**: Хибридизација је једна од кључних метода за унапређење перформанси постојећих метахеуристичких решења.
- **Хипотеза (Х3)**: Примена *NLP* метода заједно са метахеуристичким решењима постиже знатно боље резултате.

- **Хипотеза (X4)**: Алгоритам *PSO* као и његове две предложене нове верзије показују робусност на свим тестовима као и високу стабилност која одликује овај алгоритам.

1.3 Методе истраживања

Докторска теза примењује различите методе истраживања. Најпре је потребно нагласити да је аналитички метод био главна метода, те да је литература указивала на недостатак имплементација које примењују методе унапређења алгоритама примењених у овом раду. Следећа метода која је пресудна за ово истраживање јесте метода моделовања. Ова метода даје резултат две нове верзије *PSO* алгоритма, од којих је једна окренута према оптимизацији проблема поморског саобраћаја и то за класификацију и регресију, а друга верзија је моделована према потребама детекције грешака у програмском коду. Како би остварени резултати, односно остварено унапређење било верификовано, примењене су и статистичке методе заједно са компаративним методама.

Резултати истраживања су верификовани опширним тестирањем које за циљ има да прикаже робусност тестираних решења. Примењени су постојећи скупови података за све извршене експерименте.

Алгоритми поређени у оквиру једног експеримента су тестирани под истим условима што чини подешавања експеримената, скуп података, евалуационе метрике и хардвер на којим су тестови извршени. Узимајући у обзир опсежност тестова, за одређене метрике су приметни бољи резултати код основне верзије *PSO* алгоритма, који је препознатљив по стабилности. Самим тим, унапредити овај алгоритам и задржати стабилност је изазов. Поред тога што су у неком случају основне верзије биле боље, ти случајеви се могу приписати *NFL* теореми, а узимајући у обзир укупне резултате предложена решења су видно боља од основних верзија, али и других метахеуристика које су тестиране.

1.4 Научни доприноси

Доприноси науци које овај рад даје су:

- Увођење стабилног оквира за решавање важних питања у идентификацији и праћењу поморских пловила.
- Предлог за робустан механизам заснован на екстремном појачавању градијента (енгл. - *eXtreme gradient boosting* - *XGBoost*) за идентификацију и класификацију морских пловила.
- Предлог за приступ заснован на временским серијама који користи дугу краткорочну меморију (енгл. - *long short-term memory* - *LSTM*) мреже за предвиђање путање пловила
- Увођење унапређене верзије *PSO* посебно за решавање потреба за подешавање хиперпараметара за *XGBoost* и *LSTM*.

- Тумачење модела класификације са најбољим учинком коришћењем објашњивих метода вештачке интелигенције да би се боље разумео значај карактеристика када се решава овај хитан проблем.

- Метахеуристички алгоритми замењују напад грубом силом (енгл. - *brute force*) и ручни избор хиперпараметара, користећи теоријски утемељен метод за решавање проблема са великим бројем димензија.

- Ови хибридни модели доносе теоријски напредак где метахеуристички алгоритми извршавају задатке подешавања, док модели машинског учења обављају задатак класификације.

- Увођење унапређене верзије *PSO* за унапређење проблема детекције грешке у програмском коду.

- Интерпретабилност модела омогућава транспарентну оптимизацију сложених модела. Ово је кључно за поверење и одговорност модела за предвиђање грешака у софтверу, у случају да стручњаци за обезбеђење квалитета морају да потврде прогнозе модела.

- Коначно, овакав приступ представља велики помак ка аутоматизованим системима за идентификацију софтверских грешака, где се метахеуристички алгоритми могу користити за аутоматизацију многих одлука које обично захтевају људску интервенцију, чиме се смањују трошкови пројекта и повећава поверење у квалитет пројекта.

Резултати представљени у докторској дисертацији су објављени у два престижна научна часописа међународног карактера категорија М21а [7] и М22 [8]. Истраживање и резултати који су претходили овом истраживању чине још и научни радови објављени у врхунским часописима и то 1 категорије М21а [9], 4 категорије М21 [10, 11, 12, 13], 1 категорије М22 [14] и 2 категорије М24 [15, 16]. Квалитет и веродостојност резултата су верификовани од стране стручњака из области.

1.5 Структура дисертације

Представљено истраживање се односи на проблеме глобалне оптимизације применом алгоритама инспирисаних природним феноменима. Због великог броја експеримената и различитих модела који су тестирани, детаљнији опис структуре рада је представљен у даљем тексту. По завршетку уводног дела следи Глава 2 са сврхом представљања две главне групе експеримената. Подела на овај начин представља две групе проблема, поморског саобраћаја и грешака у програмском коду, али је такође битно нагласити да је на овом нивоу могуће поделити докторску дисертацију на два дела по основу врста модификација примењених на *PSO* алгоритам узимајући у обзир да оба случаја примене имају различите захтеве. Глава 3 је посвећена описивању метахеуристичких алгоритама и њихових карактеристика. Поред тога, у наведеној глави су приказани детаљни описи сваког алгоритама који је примењен у обе наведене групе експеримената. Глава 4 служи да представи остале типове решења из области вештачке интелигенције а који су примењени у овом раду и такође детаљно

описани. Глава 5 је употребљена за предлог промена на основну верзију *PSO* алгоритама и подељено је као и Глава 2. Резултати истраживања су представљени у Глави 6. Коначне закључке је могуће прочитати у Глави 7 као и усмерење за наставак истраживања предложених тема.

Претходни параграф је посвећен описивању буквалне структуре садржаја докторске дисертације. Мање опипљива врста структуре овог рада је сачињена од научних радова који су претходили овом истраживању, али и оних из којих су два главна објављена истраживања која чине ову тезу проистекла.

Глава 2

Опис проблема

Као што је већ наглашено, ово истраживање је могуће поделити на две целине. Подела се врши по природи проблема које је потребно решити применом *PSO* алгоритма. Детаљи везани за ове експерименте приказани су у следећа два поглавља. Прво су описани експерименти у којима се примењује вештачка интелигенције у предвиђању типа пловила и путање, као и други за предвиђање грешке у програмском коду.

За прву групу је примењена *LSTM* мрежа за предвиђање путање, док је *XGBoost* примењен за класификацију бродова. Предложена методологија је примењена као оптимизатор обе ове методологије, односно бустована оптимизиацијаројевима честица (енгл. - *boosted particle swarm optimization - BPSO*).

Друга група се може поделити на три целине. Прве две целине чине експерименти на два различита слоја предложеног решења. На првом слоју (енгл. - *layer 1 - L1*) је примењена конволуциона неуронска мрежа (енгл. - *convolutional neural networks - CNN*) за предвиђање грешке у програмском коду. Други слој (енгл. - *layer 2 - L2*) узима резултате примене *CNN* на цео скуп података и тренира над њима у два одвојена случаја адаптивни бустинг (енгл. - *adaptive boosting - AdaBoost*) и *XGBoost* као главне предикторе. Трећу целину чине симулације које се извршавају само на *L2*, али у комбинацији са *NLP* методама поново у два одвојена случаја са *AdaBoost* и *XGBoost*.

2.1 Експерименти предвиђања поморског саобраћаја

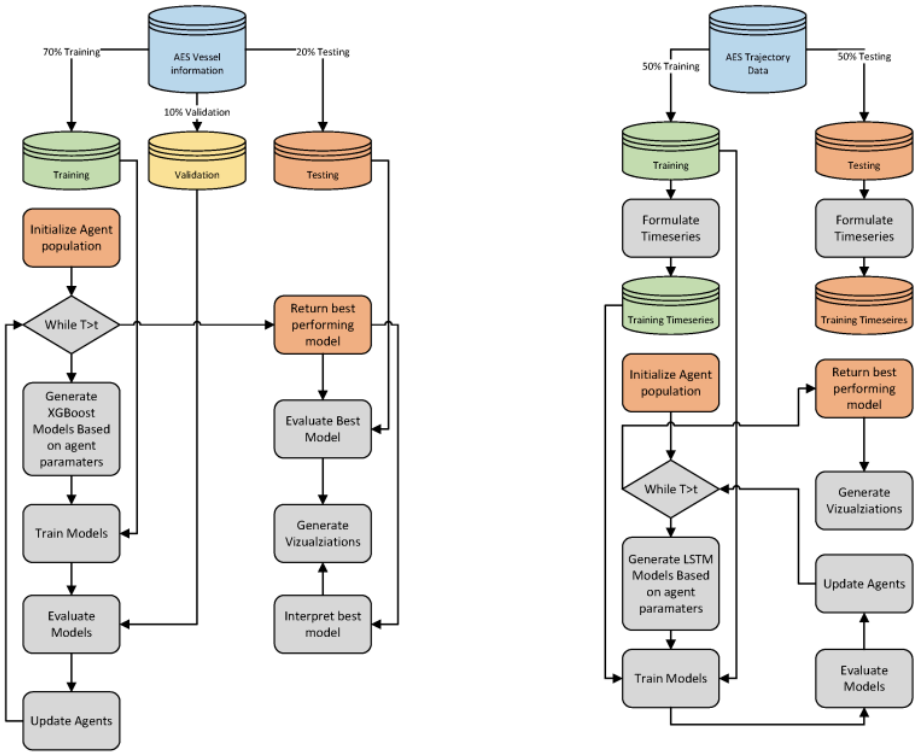
Огромна количина података се генерише приликом обављања поморског саобраћаја. Услови на отвореном мору су другачији и суровији од свакодневних видова транспорта. Из овог разлога се користи више уређаја за праћење њиховог кретања и не само то, већ су то и уређаји који захтевају софистицирани хардвер што знатно повећава трошкове. Овакав вид праћења је пре свега настао да би пловило могло да одржи курс. Кроз историју су се методе мењале те данас постоје методе којима је могуће наводити брод али и пратити његову локацију у реалном времену [17]. Није само напредовала опрема за праћење већ и методе које обрађују податке који су прикупљени. Методе анализе података тако омогућавају прецизније информације о понашању пловила. На пример, просторно груписање за примене са шумом (енгл. - *density-based spatial clustering of applications with noise - DBSCAN*) засновано на густини коришћења се примењује за моделирање понашања пловила на основу по-

датака тачака путање. Помоћу примене наведене методе могуће је уочити аномалије при путањи пловила као што су неочекивана заустављања, одступање од утврђене руте или иррегуларност брзине.

Конкретно, у контексту рибарских пловила, развијени су модели детекције аномалија у реалном времену засновани на рачунарству на ивици (енгл. - *edge computing*) [18]. Ови модели у потпуности користе информације о покретним ивичним чворовима и оближњим чворовима, односно уређајима који могу да локално обрађују податке. Ови уређаји комбинују историјски модел екстракције путање са моделом детекције аномалија на мрежи. Овај приступ се показао ефикасним у откривању аномалија рибарских пловила у реалном времену, пружајући драгоцене информације о обећавајућим локацијама за риболов и временским аномалијама.

Активности у вези са поморским путовањем тешко је пратити због њихове природе и недостатка сензора који могу да снимају податке. У већини случајева, главни систем за праћење пловила је његов глобални позициони систем (ГПС). Систем за аутоматску идентификацију (енгл. - *automatic identification system* - *AIS*) се користи за праћење путања пловила и нуди прецизније информације о рути пловидбе [19, 20, 21]. Уочени подаци пружају увид у аномалије на планираном путу, попут скретања са курса и необјашњивих препрека. То су обично знаци илегалних активности, као што је криволов. Други феномен се може објаснити тиме да рибарске посаде желе да обављају незаконите радње искључивањем транспондера повезаних са *AIS*. Проблем коришћења нерегистроване опреме и више пловила него што је дозвољено за риболов у појединим областима штети дивљини. Сва наведена понашања су штетна по животну средину и изазивају проблеме у морској флори и фауни.

Сврха овог експеримента је да се предложи ново решење засновано на *XGBoost PSO* оптимизованој *LSTM* мрежи за класификацију пловила и предвиђање путање. Метахеуристике, посебно фамилија ројева, показала се одличним оптимизаторима за НП тешке проблеме којима се сматра да припада оптимизација *LSTM*. Класификација се врши као први експеримент за идентификацију пловила, док се регресија ради током другог експеримента за предвиђање путање пловила. Коначно, циљ експеримената је радни оквир који користи хибридну верзију *PSO* алгоритма за оптимизацију *XGBoost* и *LSTM*. Експерименти класификације пловила и предвиђања путање пловила су илустровани Сликом 2.1.



Слика 2.1: Дијаграм тока предложеног радног оквира за експерименте предвиђања поморског саобраћаја.

2.2 Експерименти препознавања грешке у програмском коду

Улога софтвера у модерном друштву је одавно достигла критичну тачку где би приликом неког већег широко распрострањеног проблема последице биле на нивоу друштва [22]. Из претходне реченице се може закључити да је неопходан робустан систем за управљање грешкама. Последице софтверских дефеката могу бити тривијалне, али и опасне по живот, јер се примена софтвера креће од забаве до медицине. Интернет ствари (енгл. - *internet of things* - *IoT*) и вештачка интелигенција у великој мери доприносе овој зависности од софтвера [23], пошто све више уређаја користи софтвер који је све захтевнији. Одговорност програмера се повећава јер неки случајеви употребе као што су аутономна возила и медицина захтевају много опсежније тестирање [24]. Иако софтверски дефект у неким случајевима може бити на нивоу непријатности, чак и ти случајеви би имали користи од *SDP* [25]. Кључни допринос *SDP* у развоју софтвера је у фази тестирања. Главни принцип је посветити више пажње модулима који су склони грешкама. Такав увид у стање пројекта може омогућити програмерима да раније открију грешке или се чак припреме за њих у

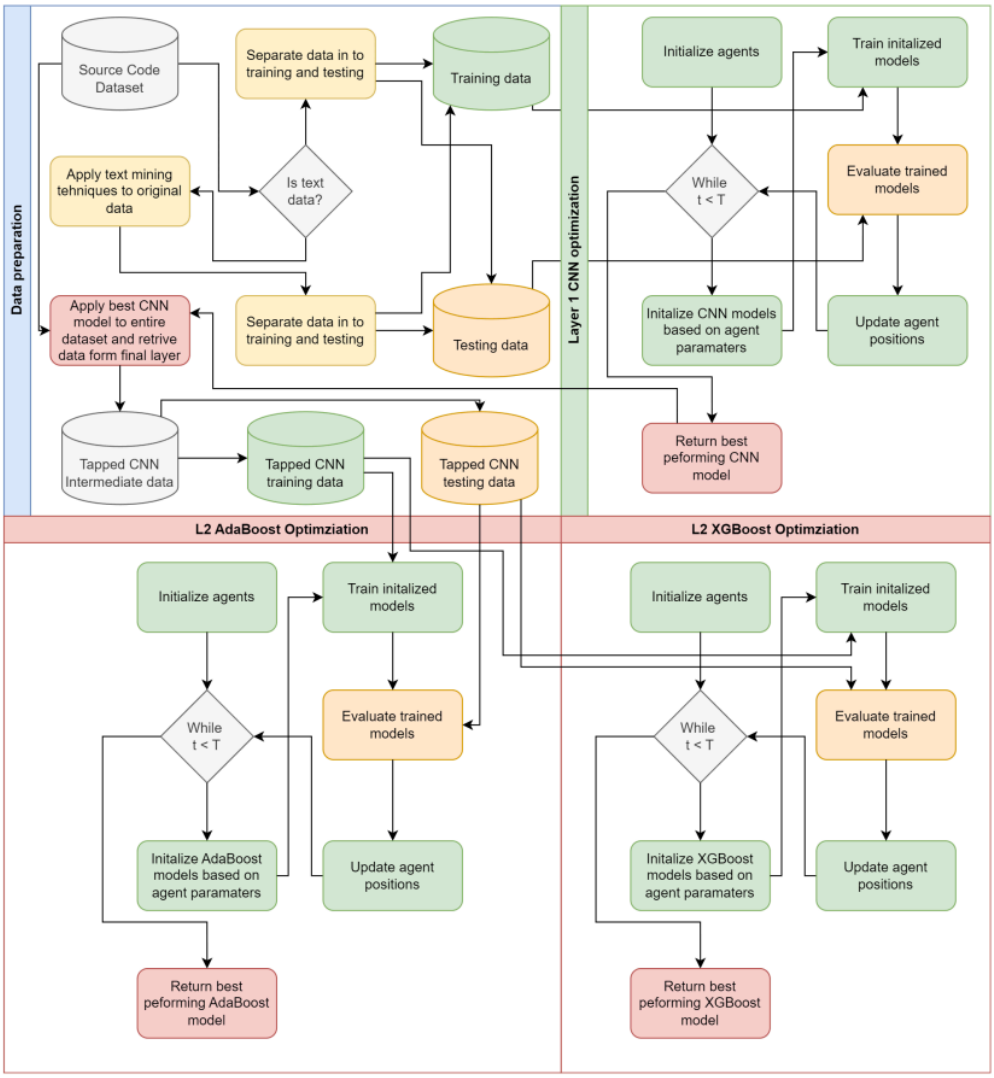
одређеним окружењима.

Процес производње софтвера назива се животни циклус развоја софтвера (енгл. - *software development life-cycle - SDLC*) током којег би требало примењивати *SDP* да би се смањио број грешака. Софтвер скоро никада није савршен и постала је уобичајена пракса да програмери објављују недовршене пројекте и раде на њима итеративно кроз ажурирања. Коришћењем конвенција за писање програмског кода и других принципа, грешке се могу свести на минимум, али никада у потпуности искоренити. Због тога је неопходан робустан систем који би помогао програмерима да пронађу грешке. Са оваквим системом, грешке би се могле открити раније што би могло спречити значајне финансијске губитке. За мерење квалитета кода израчунава се густина дефеката. Најчешћи облик таквог мерења је број дефеката на хиљаду линија кода (енгл. - *defects number per thousand lines of code - KLOC*).

Напреци вештачке интелигенције, посебно машинског учења, указују на велики потенцијал не само за различите *NLP* примене [26], већ и у другим областима, укључујући интеракцију човека и робота [27], предиктивно моделирање [28] и предвиђање временских серија [29]. Узимајући у обзир да је програмски код такође језик, овај потенцијал треба истражити и за предвиђања у вези са програмским језицима. Када се упореде природни и програмски језици могу се уочити разне сличности, али су програмски језици строжи у погледу правила писања која помажу у препознавању образаца. Процес контроле квалитета може се побољшати употребом вештачке интелигенције која може да поједностави процес откривања грешака и обезбеди бољу покривеност тестова за верификацију исправности кода. Да би се открили грешке у тексту, потребно је разумети које методе се примењују кроз *NLP* да би се то постигло. Токенизација је један од кључних концепата и њена улога је да сегментира текст у мање јединице за обраду. Свођење речи на њихове основне облике (енгл. - *stemming*), док лематизација идентификује корен речи помоћу речника и морфолошке анализе. Различити делови реченица као што су именице, глаголи и придеви идентификују се кроз процес рашчлањивања. Потенцијал *NLP* за проблем *SDP* је недовољно истражен.

Примењени експеримент користи двослојни оквир за предвиђање грешака у софтверу. Испитани су бројни модели, међутим, *CNN* су показале изванредне могућности издвајања обележја (енгл. - *feautre extraction*).

Ипак, претходни истраживачки радови [30, 31, 32] су показали да ако се густи слој *CNN* замени са *XGBoost* или *AdaBoost*, класификација се побољшава. Другим речима, *XGBoost* и *AdaBoost* класификатори обављају задатак класификације знатно боље од традиционалних густих слојева *CNN*. Ово истраживање, познати алгоритам *PSO* примењује као оптимизатор који се користи за подешавање модела за овај конкретни задатак. Основна верзија је унапређена, а модификације ће бити детаљно представљене у наредним главама. Без обзира на друге новије доступне оптимизаторе предложени алгоритам је постигао веома охрабрујуће резултате на експериментима мањег обима, па је стога изабран за додатне измене како би постигао још боље резултате за овај проблем. Експерименти предвиђања софтверске грешке су подељени у три групе за које су дијаграми токова приказани Сликом 2.2. Овде су приказани експерименти на *L1* са применом *CNN*, као и на *L2* где су одвојено примењени *AdaBoost* и *XGBoost* модели.



Слика 2.2: Дијаграм тока предложеног радног оквира за експерименте грешке у програмском коду.

Глава 3

Метахеуристичка оптимизација

Велики број процеса се може оптимизовати. Две главне поделе оптимизације се односе на то да ли је потребно пронаћи најмању грешку, односно проблем минимизације, или да ли је потребно пронаћи најбољу вредност, односно проблем максимизације. Примене ових врста оптимизације могу бити на квалитет или профит по питању максимизације или смањење трошкова и времена када је у питању минимизација. Већина случајева у којима је потребна оптимизација и где се могу постићи знатна унапређења су из области науке и инжењерства. Овакви проблеми оптимизиације су сложени и није их могуће оптимизовати ручно. Са друге стране, нема смисла примењивати захтеван софтер за оптимизовање неког процеса ако је он сам по себи једноставан и могуће је то урадити без помоћи рачунара.

Традиционалне методе при решавању НП тешких проблема не постижу задовољавајуће резултате због проблема заглављивања у локалним оптимумима. До овога долази јер комплекснији проблеми имају мноштво локалних оптимума и применом једноставнијих метода оптимизације није могуће наставити процес после локалног оптимума јер се сматра да су тада пронашле решење. Оптимизациони проблеми из стварног живота најчешће имају НП тежину те је потреба за ефикасним оптимизаторима велика.

Да би овај концепт био боље разјашњен, неопходно је да се уведе појам оптималног, што у случају примене овог рада значи да је то најбоље могуће решење. Узимајући у обзир да још увек није реално да постоји начин да се пронађе одговор на све, апроксимативни алгоритми проналазе субоптимално решење или другим речима решење најприближније оптималном.

Процес израчунавања најбољег решења при одређеним ограничењима се назива оптимизација. Могуће је поставити скоро све процесе као проблем оптимизације, те је закључак да је оптимизација једна од кључних области за даљи напредак човечанства. Три компоненте чине основне проблеме оптимизације, а то су променљива за одлучивање којих може бити више или једна, функција циља (енгл. - *objective function*) и ограничења. Променљива одлучивања је по својој природи дискретна, континуална или мешовита, односно дискретно-континуална. Функција циља је позната још и као критеријумска функција, односно функција губитака и у овом случају се ради минимизација грешке. Ограничење може бити у виду једнакости или неједнакости. Решење се сматра изводљивим (енгл. - *feasible solution*) ако променљива одлучивања задовољава ова ограничења. Проблем оптимизације се математички може представити Једначином 3.1.

$$f : \mathbb{R}^n \to \mathbb{R}, \quad x^* \in \mathbb{R}^n, \quad \forall x \in \mathbb{R}^n \ (f(x^*) \leq f(x)). \tag{3.1}$$

Тежина решавања проблема оптимизације се може одредити рачунском сложеношћу (енгл. - *computational complexity*). За проблеме класификације комплексност може бити детерминистичког полиномског времена (П), или већ описана НП тежина, односно недетерминистичког полиномског времена. Проблемима П типа се сматрају лакши проблеми, решиви у детерминистичком полиномском времену. Стога НП тешки проблеми нису решиви у детерминистичком полиномском времену, док се њихово решење може проверити у полиномском времену.

Као што је већ поменуто, проблеми са којима се људи свакодневно сусрећу су претежно јако сложени и припадају НП класи проблема. Неки од примера укључују нелинеарне, мултимодалне, сложене и проблеме са опширним простором претраге. Ове проблеме је немогуће решити у разумном временском интервалу. Традиционални алгоритми су погодни за локалну претрагу и не постижу добре резултате при глобалној оптимизацији. Овакви проблеми могу бити решени путем апроксиматвних алгоритама. Хеуристичка решења овог типа су специјализована за одређени проблем, док се метахеуристички алгоритми могу примењивати на разне проблеме оптимизације. Специфичност метахеуристичких алгоритама је да траже задовољавајуће решење у разумном временском интервалу за одређени проблем. Разумни временски интервал је онај у којем је резултат и даље релевантан, што не би био случај када би се тражило савршено, односно најбоље решење.

Термин метахеуристика се може разложити на две речи, мета и хеуристика. Прва реч на грчком значи изван, на вишем нивоу, док друга представља откривање, те скупа означавају проналажење опште стратегије за решавање неког проблема. Алгоритме из ове групе карактерише случајност, што их чини стохастичким решењима. Метахеуристичке методе готово увек карактеришу и две јасно одвојене фазе проналажења решења. Изузеци од овога шаблона су ретки, али постоје алгоритми који припадају групи метахеуристичких решења који немају јасно дефинисане фазе. Експлорација представља прву фазу, односно проналазак дела укупног простора претраге који је прикладан за интензивну претрагу. Експлоатација представља другу фазу, која се надовезује на претходну тако што у пронађеном оптималном региону интензивно тражи најбоље решење. Оваква врста решења је погодна за решавање комплексних проблема оптимизације, а да се притом могу имплементирати као и комбиновати са постојећим системима релативно једноставно. Постоје и даље поделе метахеуристичких решења, а за овај рад је најважније навести да поред *SI* постоје и еволутивни алгоритми (енгл. - *evolutionary algorithms*) [33] инспирисани управо тим принципом, као и математички инспирисани алгоритми.

3.1 Оптимизација роjевима честица

Оптимизација роjевима честица, односно *PSO* је алгоритам представљен 1995. године чији су аутори Џејмс Кенеди (енгл. - *James Kennedy*) и Расел Еберхарт (енгл. - *Russell Eberhart*) [3]. Инспирација за овај алгоритам потиче од птица и риба које се крећу у јатима. Индивидуе у популацији се сматрају честицама. Проблеми дискретне као и континуиране оптимизације су решиви овом методом. Први корак је додела почетних брзина свакој честици која је део популације. Брзине су описане са три тежине које дефинишу њихово кретање према оптималном решењу. Ове тежине су

терминална брзина, најбоља тежина остварена до сада, као и најбоља остварена тежина од стране честице у непосредној близини.

Промена вредности, односно ажурирање позиција и брзина честица (енгл. - *particle position and velocities updating*) се врши Изразом 3.2.

$$\begin{cases} \overrightarrow{v_i} \leftarrow \overrightarrow{v_i} + \overrightarrow{U}(0,\phi_1) \bigotimes (\overrightarrow{p_i} - \overrightarrow{x_i}) + \overrightarrow{U}(0,\phi_2) \bigotimes (\overrightarrow{p_g} - \overrightarrow{x_i}), \\ \overrightarrow{x_i} \leftarrow \overrightarrow{x_i} + \overrightarrow{v_i}, \end{cases} \tag{3.2}$$

у којој је Хадамардов производ $\bigotimes$, компоненте v_i су у опсегу $[-V_{max}, +V_{max}]$, и вектор који представља сваку честицу кроз све генерације униформно распоређен у интервалу $[0, \phi_i]$ је $\overrightarrow{U}(0, \phi_1)$.

Свака честица представаља могуће решење у D-димензионалном простору. Позиција честице у популацији је приказана Једначином 3.3.

$$X_i = (x_{i1},\ x_{i2},\ ...,\ x_{iD}), \tag{3.3}$$

Најбоља пронађена позиција честице је описана Једначином 3.4.

$$P_i = (p_{i1},\ p_{i2},\ ...,\ p_{iD}), \tag{3.4}$$

Брзине кретања честица су приказане Једначином 3.5.

$$V_i = (v_{i1},\ v_{i2},\ ...,\ v_{iD}). \tag{3.5}$$

Најбоље пронађено решење до сада је означено као *pbest*, а најбоље из групе као *gbest*. Позиција следеће честице се рачуна помоћу ове две вредности. Уколико се примени приступ инерционе тежине може се приказати Једначином 3.6.

$$v_{id} = W * v_{id} + c_1 * r_1 * (P_{id} - X_{id}) + c_2 * r_2 * (P_{gd} - X_{id}), \tag{3.6}$$

где је почетна брзина честице v_{id}, тренутна позиција x_{id}, фактор инерције w, релативни когнитивни утицај је c_1, утицај друштвене компоненте c_2, а r_1 и r_2 су насумични бројеви. Вредности за *pbest* и *gbest* су представљене кроз p_{id} и p_{gd}, респективно.

Фактор инерције се рачуна по Једначини 3.7.

$$w = w_{max} - \frac{w_{max} - w_{min}}{T} \cdot t, \tag{3.7}$$

где је почетна тежина w_{max}, коначна тежина w_{min}, максимални број итерација T, а тренутна итерација t.

Псеудокод *PSO* алгоритма је приказан Алгоритмом 1.

Алгоритам 1 Псеудокод *PSO*.

Иницијализација популације честица са насумичним позицијама и брзинама у D-димензионалном простору
repeat
  Евалуација жељене функције погодности за сваку честицу у D променљивих
  Поређење исхода са $pbest_i$
  if Тренутна вредност је боља од $pbest_i$ **then**
    Тренутна најбоља вредност се подешава као вредност $pbest_i$
    Вредност $\overrightarrow{p_i}$ постаје тренутна позиција $\overrightarrow{x_i}$ у D-димензионалном простору
  end if
  Додељивање индекса тренутно најбољег решења променљивој g
  Промена позиција честице по формули:

$$\begin{cases} \overrightarrow{v_i} \leftarrow \overrightarrow{v_i} + \overrightarrow{U}(0,\phi_1) \bigotimes (\overrightarrow{p_i} - \overrightarrow{x_i}) + \overrightarrow{U}(0,\phi_2) \bigotimes (\overrightarrow{p_g} - \overrightarrow{x_i}) \\ \overrightarrow{x_i} \leftarrow \overrightarrow{x_i} + \overrightarrow{v_i}, \end{cases}$$

until Критеријум задовољен

3.2 Генетски алгоритам

Генетски алгоритам (енгл. - *genetic algorithm - GA*) користи еволутивни приступ, док је главна инспирација за овај алгоритам био феномен природне селекције [33]. Процеси селекције, наслеђивања, укрштања, и мутације на нивоу ћелија су стимулисани. Творац генетских аглоритама је Давид Голдберг (енгл. - *David Goldberg*) [34]. Сеједали Мирјалили (енгл. - *Seyedali Mirjalili*) је аутор *GA*. Индивидуе из иницијализоване популације имају скуп особина које представљају хромозоме подложне променама. Популација решења се иницијализује насумично и представља итеративни процес у којем се свака итерација назива генерација. Свака индивидуа има фактор погодности (енгл. - *fitness*) којим је описан њен квалитет. На основу погодности индивидуе врши се селекција за следећу генерацију. Методом укрштања се комбинују две индивидуе. Предности сваке индивидуе се преносе у индивидуу која настаје као резултат. Процесом мутације могуће је променити карактеристике постојеће индивидуе у следећој генерацији. Псеудокод *GA* је приказан Алгоритмом 2.

Алгоритам 2 Псеудокод *GA*.

Иницијализација популације n хромозома
repeat
  Израчунавање погодности $f(i)$ за сваки хромозом i
  Генерисање нове популације
    Селекција хромозома
    Укрштање
    Мутација
    Прихватање
  Замена претходне популације
until критеријум евалуације није задовољен

3.3 Оптимизација колонијама пчела

Оптимизација колонијама пчела (енгл. - *artificial bee colony - ABC*) разликује три различите улоге у колонији [35]. Ове три колоније су радници, посматрачи, и извиђачи који утичу на процесе експлорације и експлоатације. Популација је подељена у два дела. Први део представљају мрави радници, а други посматрачи. Прва група се бави потрагом за храном, што се алгоритамски преводи у потрагу за потенцијалним решењем. Друга група одлучује који је извор хране најбољи за експлоатицју из информација добијених од прве групе. Уколико јединица пронађе извор хране из којег није могуће пронаћи даље побољшање, пчела радник мења улогу у извиђача а извор хране се замењује са насумичним. Популација јединки, односно потенцијалних решења, се генерише помоћу Једначине 3.8.

$$x_{i,j} = lb_j + rand(0,1) * (ub_j - lb_j), \tag{3.8}$$

где је j-ти параметар i-тог решења $x_{i,j}$, горња и доња граница су ub_j и lb_j за j-ти параметар, респективно.

Током сваке итерације пчеле проналаже нове изворе хране у њиховој околини на основу Једначине 3.9.

$$v_{i,j} = \begin{cases} x_{i,j} + \phi * (x_{i,j} - x_{k,j}), \ R_j < MR, \\ x_{i,j}, \ otherwise, \end{cases} \tag{3.9}$$

где је $x_{i,j}$ j-ти параметар претходног решења i, $x_{k,j}$ је j-ти параметар оближњег решења k, ϕ је насумична вредност у опсегу $[0,1]$, MR је стопа модификације која контролише конвергенцију. Погодност решења се евалуира на основу претходног и у случају да је ново решење боље убраја се у популацију. По завршетку фазе интезификације, пчеле радници обавештавају пчеле посматраче о резултатима претраге. Група посматрача одлучује који ће извор хране i бити изабран на основу пропорционе вероватноће вредности погодности и приказан Једначином 3.10.

$$p_i = \frac{fit_i}{\sum_{j=1}^{m} fit_j}, \tag{3.10}$$

где је вероватноћа извора i да буде изабран p_i, укупан број извора хране је m, а погодност је fit. Контролни параметар $limit$ је кључан за доношење одлуке о напуштању извора хране. Псеудокод алгоритма је приказан Алгоритмом 3.

Алгоритам 3 Псеудокод *ABC*.

```
Иницијализација
repeat
  Радници
  Посматрачи
  Извиђачи
  Чување најбољег решења до сада
until итерација = максимални број итерација
```

3.4 Алгоритам слепог миша

Слепи мишеви су фасцинантне животиње јер се убрајају у неколико ретких које имају способност ехолокације. Ово је такође и једини крилати сисар који има тако напредне способности. Управо ова особина је и била инспирација за стварање алгоритма слепог миша (енгл. - *bat algorithm* - *BA*) [36]. Математичке формуле на којима се заснива овај алгоритам имитирају кретање слепог миша на основу разних утицаја. На почетку извршавања алгоритма свака јединка у популацији добија насумичну брзину V_i према потенцијалном решењу x_i. Фреквенција зависи од дужине таласа и гласноће. Док тражи плен, фреквенција, гласноћа и учесталост емитовања еха варирају код слепих мишева. Процес интезификације је базиран на насумичном ходу (енгл. - *random walk*). Критеријум за проналазак решења се подешава, а фазе експлорације и експлоатације се контролишу помоћу параметара. Због једноставности, правила понашања слепих мишева су представљена на следећи начин:

- Сви слепи мишеви користе ехолокацију за осећај удаљености, али такође имају способност да некако разликују храну и баријере.
- Гласноћа може имати вредности у широком опсегу, али се претпоставља да је то између јако позитивних $\mathbf{A}_0$ и најмањих константих вредности $\mathbf{A}_{min}$.

Кретање слепих мишева је приказано следећом Једначином 3.11.

$$x_i^t = x_i^{t-1} + v_i^t, \tag{3.11}$$

где је тренутно решење x_i^{t-1}, а ново и ажурирано решење је x_i^t у t-тој итерацији за i-то решење. Брзина је представљена као v_i^t и рачуна се за временски корак t по Једначини 3.12.

$$v_i^t = v_i^{t-1} + (x_i^{t-1} - x_*)f_i, \tag{3.12}$$

где је глобално најбоља позиција представљена као x_*, а f_i приказује учесталост i-тог слепог миша. Учесталост решења је униформно представљена између минималне и максималне учесталости описане Једначином 3.13.

$$f_i = f_{min} + (f_{max} - f_{min})\beta, \tag{3.13}$$

где су f_{min} и f_{max} минимална и максимална учесталост, респективно, а β је насумични број у опсегу $[0, 1]$. Насумични ход мења тренутно најпогоднije решење и усмерава алгоритам у процесу експлорације што је и описано Једначином 3.14.

$$x_{new} = x_{old} + \epsilon A^t, \tag{3.14}$$

где је просечна гласноћа слепих мишева представљена као A^t, а фактор скалирања са насумичним вредностима у опсегу $[0, 1]$ је ϵ. Када слепи миш пронађе плен, гласноћа се мења на основу Једначине 3.15.

$$A_i^t = \alpha A_i^{t-1},\ r_i^t = r_i^0[1 - exp(-\gamma t)]. \tag{3.15}$$

Промену вредности параметара из Једначине 3.15 је могуће описати Изразом 3.16.

$$A_i^t \to 0,\ r_i^t \to r_i^0,\ \text{while } t \to \infty, \tag{3.16}$$

где A_i^t приказује гласноћу i-тог слепог миша у итерацији t, а r је учесталост емитовања еха. Вредности α и γ представљају константе.

Псеудокод *BA* је приказан Алгоритмом 4.

Алгоритам 4 Псеудокод *BA*.

Цильна функција $f(x)$
Иницијализација популације, v_i, r_i и A_i, вредности дефинишу учесталост еха (f_i) у x_i, максимални број итерација ($MaxIter$), а бројач итерације (t) је подешен на 0

while $t < MaxIter$ **do**
  Генерисање нове генерације решења подешавањем позиције по Једначини 3.11, брзине по Једначини 3.12 и учесталости по Једначини 3.13
  if $rand > r_i$ **then**
    Избор најпогоднијег решења
    Насумичини ход
  end if
  if $(p_i < A_i)$ и $f(x_i) < f(x_*)$ **then**
    Прихватање новог решења
    A_i смањење и r_i повећање
  end if
  Чување најбољих решења до сада, x_*
end while
Враћање најбољег решења

3.5 Алгоритам оптимизације китова

Китови се сматрају за јако интелигентне животиње због вретенастих ћелија које деле са људима. Ове ћелије су заслужне за расуђивање, емоције и слична друштвена понашања. Инспирација за алгоритам оптимизације китова (енгл. - *whale optimization algorithm* - *WOA*) је начин на који ове животиње лове који се назива храњење методом мехурића (енгл. - bubble net feeding) [37]. Грбави китови лове тако што емитују звук високе фреквенције који усмерава плен ка површини а затим плен окружују мрежом од мехурића које испуштају како плен не би побегао. Уколико лове сами примењују методу дуплих петљи (енгл. - *double loops*), а уколико су у групи могу користити и методу усправних спирала (енгл. - *upward-spirals*). Дуплу петљу китови започињу тако што се спусте на дубину од око 12 метара. Метода усправних спирала се дели у три фазе: мрежа мехурића, ударац репом о површину и израњање ради хватања заробљене рибе. Овај алгоритам адаптира процес храњења мехурићама као стратегију оптимизације.

3.5.1 Окруживање плена

Тренутно најбоље решење је плен, а позиција јединке у популацији се ажурира на основу Једначине 3.17 и Једначине 3.18.

$$\overrightarrow{D} = |\overrightarrow{C} \cdot \overrightarrow{X^*}(t) - \overrightarrow{X}(t)|, \tag{3.17}$$

$$\overrightarrow{X}(t+1) = \overrightarrow{X^*}(t) - \overrightarrow{A} \cdot \overrightarrow{X}, \tag{3.18}$$

где су коефицијенти вектора $\overrightarrow{A}$ и $\overrightarrow{C}$, тренутна итерација је t, позиција најбољег решења је $\overrightarrow{X^*}(t)^2$, док $\cdot$ и $||$ приказују Хадамардов производ и апсолутну вредност у том редоследу.

Једначина 3.19 и Једначина 3.20 служе за израчунавање вредности коефицијента вектора и тиме балансирају фазе.

$$\overrightarrow{A} = 2\overrightarrow{a} \cdot \overrightarrow{r} - \overrightarrow{a}. \tag{3.19}$$

За случај у којем је $A \geq 1$ биће извршена фаза експлорације, док за $A < 1$ ће бити извршена фаза експлоатације. Уколико се параметар C подеси на насумичну вредност у опсегу $[0, 2]$ експлоатација може дати боље резултате.

$$\overrightarrow{C} = 2 \cdot \overrightarrow{r}, \tag{3.20}$$

где $\overrightarrow{a}$ линеарно опада у опсегу $[2, 0]$ током обе фазе, а насумчни вектор $\overrightarrow{r}$ је у опсегу $[0, 1]$. Коначни број итерација је представљен као I_{max}, док је тренутна итерација обележена са t. Параметар $\overrightarrow{a}$ се ажурира по правилу $\overrightarrow{a} = 2(1 - t/I_{max})$.

3.5.2 Напад методом мреже мехурића

Вектор коефицијента A је подешен у опсегу $[-1, 1]$ и вредност се линеарно смањује. Позиција као резултат наведеног би требало да буде између тренутне и најбоље позиције. Адаптација кретања грбавог кита је приказана Једначином 3.22, док Једначина 3.21 рачуна размак између тренутне локације и тренутно најбоље локације.

$$\overrightarrow{D'} = |\overrightarrow{X^*}(t) - \overrightarrow{X}(t)|, \overrightarrow{X}, \tag{3.21}$$

$$\overrightarrow{X}(t+1) = \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X^*}(t), \tag{3.22}$$

где константа b дефинише логаритамски спирални облик, док је l насумичан број у опсегу $[-1, 1]$.

Китови истовремено пливају око свог плена смањујући кругове који праве и крећу се по спиралној путањи. Због овога је потребно применити оба ова приступа алгоритамски. Вероватноћа од 50% је додељена овим методама што је описано у Једначини 3.23.

$$\overrightarrow{X}(t+1) = \begin{cases} \overrightarrow{X^*}(t) - \overrightarrow{A} \cdot \overrightarrow{D}, \text{ if } p < 0.5, \\ \overrightarrow{X}(t+1) = \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X^*}(t), \text{ if } p \geq 0.5, \end{cases} \tag{3.23}$$

где је p насумичан број у опсегу $[0, 1]$.

3.5.3 Потрага за пленом

Претрага за храном се такође врши преко методе смањујућих кругова. Да би се проширио простор претраге у којем јединка из популације у алгоритму тражи извор хране потребно је имплементирати насумичност. Ово понашање је моделовано као $|\overrightarrow{A}| > 1$, док је најбоље тренутно решење $\overrightarrow{X^*}(t)$ замењено са $\overrightarrow{X_{rand}}$. Математички приказ описаног је приложен Једначином 3.24 и Једначином 3.25.

$$\overrightarrow{D} = |\overrightarrow{C} \cdot \overrightarrow{X_{rand}} - \overrightarrow{X}(t)|, \tag{3.24}$$

$$\overrightarrow{X}(t+1) = \overrightarrow{X_{rand}} - \overrightarrow{A} \cdot \overrightarrow{D}, \tag{3.25}$$

Псеудокод *WOA* алгоритма је приказан Алгоритмом 5.

Алгоритам 5 Псеудокод *WOA*.

Иницијализација популације $X_i (i = 1, 2, ...N)$
Израчунавање погодности и идентификација најбољег решења $\overrightarrow{X^*}(t)$
repeat
  for $k \leftarrow 1$ до M (број китова) **do**
    Ажурирање $a, A, C, l,$ и p
    if $p < 0.5$ **then**
      if $|A| < 1$ **then**
        Ажурирање $\overrightarrow{D}$ једначином $\overrightarrow{D} = |\overrightarrow{C} \cdot \overrightarrow{X^*}(t) - \overrightarrow{X}(t)|$ и $\overrightarrow{X}$ једначином $\overrightarrow{X}(t+1) = \overrightarrow{X^*}(t) - \overrightarrow{A} \cdot \overrightarrow{X}$
      else
        Одредити $\overrightarrow{X_{rand}}$ и $\overrightarrow{D}$ по једначини $\overrightarrow{D} = |\overrightarrow{C} \cdot \overrightarrow{X_{rand}} - \overrightarrow{X}(t)|$
        Ажурирање $\overrightarrow{X}$ по једначини $\overrightarrow{X}(t+1) = \overrightarrow{X_{rand}} - \overrightarrow{A} \cdot \overrightarrow{D}$
      end if
    else
      Ажурирање $\overrightarrow{D}$ по једначини $\overrightarrow{D'} = |\overrightarrow{X^*}(t) - \overrightarrow{X}(t)|, \overrightarrow{X}$ и $\overrightarrow{X}$ по једначини $\overrightarrow{X}(t+1) = \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X^*}(t)$
    end if
  end for
  Израчунавање погодности сваког решења
  Ажурирање $X^*(t)$ најбољих решења
  Инкремент индекса итерације $t = t + 1$
until $t > I_{max}$ или док се критеријум заустављања не постигне
Излаз: најбоља погодност и позиција

3.6 Оптимизација Харисових јастребова

Алгоритам оптимизације Харисових јастребова (енгл. - *Harris hawk optimization - HHO*) је препознатљив по добром механизму експлорације [38]. Харисов јастреб захваљујући одличном виду прати и идентификује плен. Једини услов јесте да је плен видљив, односно да није сакривен. Да би ухватиле плен, ове птице морају да чекају и осматрају пустињске пределе које насељавају.

Јастребови су представљени као потенцијална решења у овом алгоритму, док у свакој итерацији најбоље решење представља жељени или скоро оптимални плен. Јастреб осматра поље претраге са насумичне локације и користи две различите технике да идентификује плен. Позицију бирају на основу осталих чланова породице, али и плена, како би били довољно близу да спроведу напад. Параметар q контролише промене између стратегија са једнаком вероватноћом. Случај у којем је вредност $q \geq 0.5$ извршиће позиционирање по Једначини 3.26.

$$X(t+1) = X_{rand}(t) - r_1 \left| X_{rand}(t) - 2r_2 X(t) \right|, \tag{3.26}$$

док у супротном за услов $q < 0.5$ се извршава Једначина 3.27.

$$X(t+1) = X_{rabbit}(t) - X_m(t) - r_3(LB + r_4(UB - LB)), \tag{3.27}$$

где је позиција јастреба у следећој итерацији представљена као $X(t+1)$, зец, односно плен, је представљен са $X_{rabbit}(t)$, $X(t)$ описује векторски положај јастреба, r_1, r_2, r_3, r_4, и q представљају насумичне бројеве у опсегу $[0, 1]$ и ажурирају се у свакој итерацији, доња граница променљивих је LB, а горња UB и насумично изабрани јастреб из тренутне позиције је $X_{rand}(t)$. Просечна позиција јастреба се рачуна по Једначини 3.28.

$$X_m(t) = \frac{1}{N} \sum_{i=1}^{N} X_i(t), \tag{3.28}$$

где је $X_i(t)$ позиција јастреба у свакој итерацији t и N је укупни број јастребова.

Уводи се појам енергије плена који је индикатор за промену фаза описан Једначином 3.29.

$$E = 2E_0(1 - \frac{t}{T}), \tag{3.29}$$

где је максимални број итерација по независном покретању T, а иницијална енергија плена E_0 је у опсегу $[-1, 1]$.

Фаза експлоатације представља напад јастреба и стратегије којима се плен опкољава. Граница за пасивније нападе је $|E| \geq 0.5$, док је за агресивније $|E| < 0.5$. Уколико плен успешно бежи, јастребови ће га окружити док га не исцрпе што је и приказано у Једначини 3.30.

$$X(t+1) = \Delta X(t) - E|JX_{best}(t) - X(t)|, \tag{3.30}$$

где је $\Delta X(t)$ векторска разлика најбољег решења, односно плена и тренутног решења у итерацији t приказана Једначином 3.31.

$$\Delta X(t) = X_{best}(t) - X(t). \tag{3.31}$$

Стратегија којом плен бежи је насумично понашање које је контролисано параметром J који се разликује у итерацијама приказано у Једначини 3.32.

$$J = 2(1 - r_5), \tag{3.32}$$

где је насумична вредност r_5 у интервалу $[0, 1]$. За $r \geq 0.5$ и $|E| < 0.5$, плен се сматра измореним и агресивније стратегије се примењују, а у том случају се позиције рачунају по Једначини 3.33.

$$X(t+1) = X_{best}(t) - E|\Delta X(t)|. \tag{3.33}$$

Уколико плен и даље не одустаје, јастребови ће применити другу цик-цак (енгл. - *zig-zag*) стратегију напада приказану у Једначини 3.34.

$$Y = X_{best}(t) - E|JX_{best}(t) - X(t)|, \tag{3.34}$$

а кретање је описано Једначином 3.35.

$$Z = Y + S \times LF(D), \tag{3.35}$$

где је димензија проблема D, насумични вектор величине $1 \times D$ је S, а Леви летови (енгл. - *Lévy flights*) LF се рачунају Једначином 3.36.

$$LF(x) = 0.01 \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}},\ \sigma = (\frac{\Gamma(1+\beta) \times sin(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2}) \times \beta \times 2^{(\frac{\beta-1}{2})}})^{\frac{1}{\beta}}, \tag{3.36}$$

Сходно томе, механизам за ажурирање позиција је описан у Једначини 3.37.

$$X(t+1) = \begin{cases} Y, \text{ if } F(Y) < F(X(t)), \\ Z, \text{ if } F(Z) < F(X(t)), \end{cases} \tag{3.37}$$

где су Y и Z израчунати путем Једначине 3.34 и Једначине 3.35.

На крају, за случај $r < 0.5$ и $|E| < 0.5$ плен се сматра измореним и агресивнији напади уз узастопна прогресивна зарањања се примењују. Раздаљина између мете пре њеног хватања је описана Једначином 3.38.

$$X(t+1) = \begin{cases} Y, \text{ if } F(Y) < F(X(t)), \\ Z, \text{ if } F(Z) < F(X(t)), \end{cases} \tag{3.38}$$

где је Y описано Једначином 3.39, а Z Једначином 3.40.

$$Y = X_{best}(t) - E|JX_{best}(t) - X_m(t)|, \tag{3.39}$$

$$Z = Y + S \times LF(D). \tag{3.40}$$

Псеудокод *HHO* је приказан Алгоритмом 6.

Алгоритам 6 Псеудокод *HHO*.

Улази: Величина популације N и максимални број итерација T
Излази: Позиција плена и његова погодност
Иницијализација насумичне популације $X_i (i = 1, 2, \ldots, N)$
while критеријум заустављања није достигнут **do**
  Израчунавање погодности јастреба
  Подеси X_{rabbit} као позицију плена (најбоља локација)
  for сваког јастреба (X_i) **do**
    Ажурирање почетне енергије E_0 и снагу скока J
    $E_0 = 2rand() - 1,\ J = 2(1 - rand())$
    Ажурирање E једначином 3.29
    if ($|E| \geq 1$) **then**
    Ажурирање вектора позиције по Једначини 3.26 и Једначини 3.27
    end if
    if ($|E| < 1$) **then**
      if ($r \geq 0.5$ и $|E| \geq 0.5$) **then**
      Ажурирање вектора позиције Једначином 3.30
      else if ($r \geq 0.5$ и $|E| < 0.5$) **then**
      Ажурирање вектора позиције Једначином 3.33
      else if ($r < 0.5$ и $|E| \geq 0.5$) **then**
      Ажурирање вектора локације Једначином 3.37
      else if ($r < 0.5$ и $|E| < 0.5$) **then**
      Ажурирање вектора локације Једначином 3.38
      end if
    end if
  end for
end while
Return X_{rabbit}

3.7 Алгоритам оптимизације шимпанзи

Шимпанзе су најближи рођаци у животињском царству људима и насељавају искључиво подручје Африке. Животиње које имају бољи однос мозга и тела (енгл. - *brain to body ratio* - *BBR*) се сматрају интелигентнијим, а шимпанзе имају овај однос најсличнији људима. Људи и шимпанзе деле дезоксирибонуклеинске киселине (ДНК) јер потичу од истог претка, великог човеколиког мајмуна. Између шимпанзе и човека поклапање ДНК је близу 99% [39].

Шимпанзе живе и крећу се на специфичан начин у друштву са процесима раздвајања и спајања (енгл. - *fission-fusion society*) који се може упоредити са процесима нуклеарне фисије и фузије. Овакво друштво је препознатљиво по променљивој величини и променљивим члановима у зависности од подручја кроз које група прође, као и од протеклог времена. Узимајући у обзир да се чланови група унутар друштва мењају и комбинују са осталим групама прво је предложена стратегија претраге независне групе у којој свака шимпанза независно врши претрагу. Као и људи, шимпанзе се разликују по способностима и интелигенцији. Алгоритам оптимизације шимпанзи (енгл. - *chimp optimization algorithm* - *ChOA*) препознаје четири врсте шимпанзи

у друштву и то су покретач (енгл. - *driver*), блокер (енгл. - *barrier*), гонич (енгл. - *chaser*) и нападач (енгл. - *attacker*). Покретачи прате плен и вребају прилику за напад. Блокери служе да препрече пут својим телом. Гоничи јуре за пленом, а нападачи предвиђају руту бекства плена и покушавају да плен усмере на ниже гране. Шимпанзе имају друштвене мотиве за лов као што су већи удео у плену, али и паренье. Да би шимпанза била нападач неопходно је да буде интелигентнија од просека и самим тим нападачи добијају већи комад хране коју касније могу користити и за размену. Сексуална мотивација чини последњу фазу лова хаотичном јер шимпанзе могу мењати улоге током лова по потреби. Фаза експлорације је моделирана са прве три групе шимпанзи, док је фаза експлоатације представљена само са нападачима.

Усмеравање и гоњење плена се рачунају следећом Једначином 3.41.

$$d = |c \cdot x_{\text{prey}}(t) - m \cdot x_{\text{chimp}}(t)|, \tag{3.41}$$

где се раздаљина шимпанзе и плена рачуна узимајући у обзир хаотичне факторе приказане са m и вектор коефицијента v.

Позиција шимпанзе се рачуна помоћу следеће Једначине 3.42.

$$x_{\text{chimp}}(t+1) = x_{\text{pray}}(t) - a \cdot d, \tag{3.42}$$

где се позиција шимпанзе рачуна на основу раздаљине и вектора коефицијента a.

Коефицијент a осликава фактор интензитета претраге описан Једначином 3.43.

$$a = 2 \cdot f \cdot r_1 - f, \tag{3.43}$$

где се вредност f смањује кроз итерације са циљем балансирања фаза експлорације и експлоатације.

Коефицијент c осликава утицај на позицију плена по Једначини 3.44.

$$c = 2 \cdot r_2. \tag{3.44}$$

Позиција нападача је одређена Једначином 3.45.

$$d_{\text{Attacker}} = |c_1 \cdot x_{\text{Attacker}} - m_1 \cdot x|. \tag{3.45}$$

Позиција блокера је одређена Једначином 3.46.

$$d_{\text{Barrier}} = |c_2 \cdot x_{\text{Barrier}} - m_2 \cdot x|. \tag{3.46}$$

Позиција гонича је одређена Једначином 3.47.

$$d_{\text{Chaser}} = |c_3 \cdot x_{\text{Chaser}} - m_3 \cdot x|. \tag{3.47}$$

Позиција покретача је одређена Једначином 3.48.

$$d_{\text{Driver}} = |c_4 \cdot x_{\text{Driver}} - m_4 \cdot x|, \tag{3.48}$$

где су [47] $x_1 = x_{\text{Attacker}} - a_1(d_{\text{Attacker}}), x_2 = x_{\text{Barrier}} - a_2(d_{\text{Barrier}}), x_3 = x_{\text{Chaser}} - a_3(d_{\text{Chaser}})$ и $x_4 = x_{\text{Driver}} - a_4(d_{x_{\text{Driver}}})$.

Четири најбоље позиције се чувају на основу којих се одређују даље позиције по Једначини 3.49.

$$x(t+1) = \frac{x_1 + x_2 + x_3 + x_4}{4}. \tag{3.49}$$

Хаотични метод за израчунавање позиција агената су описане Једначином 3.50.

$$x_{\text{chimp}}(t+1) = \begin{cases} x_{\text{chimp}}(t) - a \cdot d_{\text{chimp}}, & \mu < 0.5, \\ \text{Chaotic\_value}, & \mu \geq 0.5, \end{cases} \tag{3.50}$$

где насумични параметар μ одређује да ли ће бити примењен основни метод *ChOA* за ажурирање позиција или предложена хаотична стратегија.

Псеудокод *ChOA* је приказан Алгоритмом 7.

Алгоритам 7 Псеудокод *ChOA*.

Иницијализација популације шимпанзи $x_i (i = 1, 2, \ldots, n)$
Иницијализација f, m, a и c
Рачунање позиције сваке шимпанзе
Подела шимпанзи на независне групе
Рачунање погодности шимпанзе
$x_{Attacker}$ = најбоља јединка
x_{Chaser} = друга најбоља јединка
$x_{Barrier}$ = трећа најбоља јединка
x_{Driver} = четврта најбоља јединка
while t < максимум итерација **do**
  for сваку шимпанзу **do**
    Извлачење групе индивидуалне шимпанзе
    Примена групне стратегије за ажурирање f, m и c
    Употреба f, m, и c за рачунање a и затим d
  end for
  for сваку шимпанзу која тражи **do**
    if $\mu < 0.5$ **then**
      if $|a| < 1$ **then**
        Ажурирање позиције по једначини:

$$x_{\text{chimp}}(t+1) = x_{\text{chimp}}(t) - a \cdot d_{\text{chimp}}(t)$$

      else if $|a| > 1$ **then**
        Одабир насумичног агента
      end if
    else if $\mu > 0.5$ **then**
      Ажурирање позиције путем хаотичне мапе
    end if
  end for
  Ажурирање f, m, a и c
  Ажурирање x_{Attacker}, x_{Driver}, x_{Barrier}, x_{Chaser}
  $t = t + 1$
end while
return x_{Attacker}

3.8 Претрага променљивог суседства

Претрага променљивог суседства (енгл. - *variable neighborhood search - VNS*) је локална стратегија претраге која примењује принципе комбинаторике са циљем оптимизације. Основни принцип алгоритма је систематична промена суседства. Овај алгоритам од других локалних претрага разликује то што не прати путању. Уместо овога, метода истражује све даља и даља суседства. Промена суседства се извршава када је остварен напредак.

Основна верзија *VNS* типично извршава локалну претрагу или процедуру мућкања (енгл. - *shake procedure*) наизменично док се не задовољи критеријум заустављања. Локална претрага се користи да се пронађе локални минимум, док процедура мућкања побољшава диверзификацију. Општа верзија *VNS* користи итеративни процес који траје док сва суседства нису евалуирана. Локална претрага започиње из насумично генерисане тачке. Уколико је следеће евалуирано решење боље, претрага прелази у то решење и наставља да тражи у истом суседству, док у супротном прелази у следеће суседство.

Процедура мућкања насумично бира суседство $\mathcal{N}_l(x)$, из простора претраге $\mathcal{N} = \mathcal{N}_1, ..., \mathcal{N}_{k_{max}}$ где је свако $\mathcal{N}$ мапира x решење у суседство. Редослед у популацији одлучује које ће следеће суседство бити евалуирано. Промена суседства може бити извршена са више различитих метода. Неке од њих су секвенцијални корак, где се суседство ревидира уколико је пронађено боље решење, кружни корак, где се пролази кроз листу решења без обзира на побољшања и цевни корак, где се претрага наставља у истом суседству докле год се проналази боље решење.

Локална суседства $\mathcal{N}(x)$ тренутног решења x се претражују у свакој итерацији и мења се кроз процес унапређења са бољим решењем x'. Стратегије првог и најбољег унапређења се најчешће користе. Стратегија првог бира ново решење чим је оно и пронађено, док стратегија најбољег бира након евалуације целе популације.

Псеудокод *VNS* алгоритма је приказан у Алгоритму 8.

Алгоритам 8 Псеудокод *VNS*.

Улаз: x, $k_{\max}$, $t_{\max}$,
$k \leftarrow 1$
repeat
    $x' \leftarrow \text{Shake}(x, k)$
    $x'' \leftarrow \text{BestImprovement}(x')$
    $x \leftarrow \text{NeighbourhoodChange}(x, x'', k)$
    $k \leftarrow k + 1$
until $k = k_{\max}$
Излаз: x

3.9 Оптимизација хиперболичким синусом и косинусом

Мотивација за оптимизацију хиперболичким синусом и косинусом (енгл. - *sinh cosh optimizer - SCHO*) је балансирање фаза експлорације и експлоатације [40]. Из овог разлога су ове две фазе подељене на још две фазе. Алгоритми инспирисани мате-

матичким функцијама нису многобројни, али синус косинус алгоритам (енгл. - *since cosine algorithm - SCA*) и аритметичка оптимизација (енгл. - *arithmetic optimization algorithm - AOA*) су алгоритми вредни помена због високих перформанси које пружају.

Две карактеристике из хиперболичких функција су примењене у *SCHO*. Хиперболички косинус увек има вредност већу од један, што је у алгоритму примењено као граница у експлорацији и експлоатацији. Са друге стране, хиперболички синус има границе између $[-1, 1]$ и овај механизам се користи за унапређење решења у обе фазе. Најбитнији делови овог алгоритма су две фазе експлорације, две фазе експлоатације, стратегија ограничене претраге и механизам за промену фаза.

Основна популација изгледа као у Једначини 3.51.

$$X = \begin{bmatrix} x_{1,1} & \dots & x_{1,j} & x_{1,dim-1} & x_{1,dim} \\ x_{1,2} & \dots & x_{2,j} & \dots & x_{2,dim} \\ \dots & \dots & x_{i,j} & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N-1,1} & \dots & x_{N-1,j} & \dots & x_{N-1,dim} \\ x_{N,1} & \dots & x_{N,j} & x_{N,dim-1} & x_{N,dim} \end{bmatrix}, \tag{3.51}$$

Где су потенцијална решења представљена као X и добијају се преко једначине $X = rand(N, dim) \times (ub - lb) + lb$, где су насумичне вредности у опсегу $[0, 1]$, а ub и lb су респективно горња и доња граница, i-то решење на j-тој позицији је $x_{i,j}$, број кандидата за најбоље решења је N и број димензија је dim.

Промену фаза врши Једначина 3.52.

$$T = floor(\frac{Max_{iteration}}{ct}), \tag{3.52}$$

где је максимални број итерација означен као $Max_{iteration}$, $floor$ представља функцију заокруживања из *MATLAB*-а и прелазна гранична тачке је ct.

Механизам ажурирања позиција у првој фази експлорације је описан Једначином 3.53.

$$X_{(i,j)}^{t+1} = \begin{cases} X_{best}^{(j)} + r_1 \times W_1 \times X_{(i,j)}^{t}, & r_2 > 0.5, \\ X_{best}^{(j)} - r_1 \times W_1 \times X_{(i,j)}^{t}, & r_2 < 0.5, \end{cases} \tag{3.53}$$

у којој је тренутна итерација представљена као t, j-та позиција решења i у тренутној и следећој итерацији је представљена као $X_{(i,j)}^{t}$ и $X_{(i,j)}^{t+1}$, респективно, најбоље решење до сада је $X_{best}^{(j)}$, насумични бројеви су представљени као r_1 и r_2 у интервалу $[0, 1]$, а коефицијент тежине $X_{(i,j)}^{t}$ је приказан као W_1 и описан Једначином 3.54.

$$W_1 = r_3 \times a_1 \times (cosh\, r_4 + u \times sinh\, r_4 - 1), \tag{3.54}$$

где је монотоно опадајућа функција представљена као a_1 и израчунава се као $a_1 = 2 \times (-1.3 \times \frac{t}{Max_{iteration}} + m)$, где је m коефицијент контроле тачности експлорације. Насумични бројеви у интервалу $[0, 1]$ су r_3 и r_4, док је коефицијент високе осетљивости за контролу тачности експлорације u.

Друга фаза експлорације зависи од најбољег решења пронађеног јер се нова решења траже без усмерења. Механизам ажурирања позиција у овој фази је приказан у Једначини 3.55.

$$X_{(i,j)}^{t+1} = \begin{cases} X_{(i,j)}^{t} + |\varepsilon \times W_2 \times X_{best}^{(j)} - X_{(i,j)}^{t}|, & r_5 > 0.5, \\ X_{(i,j)}^{t} - |\varepsilon \times W_2 \times X_{best}^{(j)} - X_{(i,j)}^{t}|, & r_5 < 0.5, \end{cases} \tag{3.55}$$

где је ε изузетно мали број, W_2 је коефицијент тежине за $X_{best}^{(j)}$ који је представљен следећом једначином 3.56.

$$W_2 = r_6 \times a_2, \tag{3.56}$$

где је r_6 насумичан број у опсегу $[0, 1]$, а друга монотно опадајућа функција је a_2 и приказана је Једначином 3.57.

$$a_2 = 2 \times (-\frac{t}{Max_{iteration}} + n), \tag{3.57}$$

где је коефицијент који контролише тачност експлорације n.

Прва фаза експлоатације ажурира позиције по Једначини 3.58.

$$X_{(i,j)}^{t+1} = \begin{cases} X_{best}^{(j)} + r_7 \times W_3 \times X_{(i,j)}^{t}, & r_8 > 0.5, \\ X_{best}^{(j)} - r_7 \times W_3 \times X_{(i,j)}^{t}, & r_8 < 0.5, \end{cases} \tag{3.58}$$

где су r_7 и r_8 насумични бројеви опсега $[0, 1]$, а коефицијент тежине прве фазе експлоатације је W_3 и описан је у следећој Једначини 3.59.

$$W_3 = r_9 \times a_1 \times (cosh\, r_{10} + u \times sinh\, r_{10}), \tag{3.59}$$

где су r_9 и r_{10} насумични бројеви опсега $[0, 1]$.

Друга фаза експлоатације врши дубоку претрагу најбоље остварених решења до сада. Позиције у популацији се ажурирају по Једначини 3.60.

$$X_{(i,j)}^{t+1} = X_{(i,j)}^{t} + r_{11} \times \frac{sinh\, r_{12}}{cosh\, r_{12}} |W_2 \times X_{best}^{(j)} - X_{(i,j)}^{t}|, \tag{3.60}$$

где су r_{11} и r_{12} насумични бројеви оспега $[0, 1]$.

Стратегија ограничене претраге је приказана Једначином 3.61.

$$BS_{k+1} = BS_k + floor(\frac{Max_{iteration} - BS_k}{\alpha}), \tag{3.61}$$

где су позитивне целобројне вредности у опсегу $[1, k]$, α је осетљив параметар који утиче на фазе експлорације и експлоатације, а BS_1 се израчунава Једначином 3.62.

$$BS_1 = floor(\frac{Max_{iteration}}{\beta}), \tag{3.62}$$

где β представља контролну вредност за отпочињање стратегије.

Горња граница стратегије претраге се израчунава по Једначини 3.63.

$$ub_k = X_{best}^{(j)} + (1 - \frac{t}{Max_{iteration}}) \times |X_{best}^{(j)} - X_{second}^{(j)}|. \tag{3.63}$$

Доња граница стратегије претраге се израчунава по Једначини 3.64.

$$lb_k = X_{best}^{(j)} - (1 - \frac{t}{Max_{iteration}}) \times |X_{best}^{(j)} - X_{second}^{(j)}|, \tag{3.64}$$

где су ограничења простора претраге ub_k и lb_k, доња и горња граница респективно, а субоптималном решење на j-тој позицији је $X_{second}^{(j)}$.

Механизам претраге је описан у Једначини 3.65.

$$A = \left(p - q \times \left(\frac{t}{Max_{iteration}} \right)^{\left(\frac{\cosh \frac{t}{Max_{iteration}}}{\sinh \frac{t}{Max_{iteration}}} \right)} \right) \times r_{13}, \tag{3.65}$$

где је r_{13} насумични број у опсегу $[0, 1]$, а p и q балансирају фазе.

Псуедокод *SCHO* је приказан у Алгоритму 9.

Алгоритам 9 Псеудокод *SCHO*.

Улаз: Параметри ct, T, BS_1, u, m, n, α, β, p, q
Излаз: Најбоље решење $Best(X)$
Иницијализација параметара ct, T, BS_1, u, m, n, α, β, p, q
Насумична иницијализација позиција потенцијалних решења: $X : i, i = 1, \ldots, N$
$t \leftarrow 0$
while $t <$ Max_iteration **do**
  Израчунавање погодности потенцијалних решења(X)
  Потрага за до сада оптималним решењем
  for $i = 1$ to N **do**
    for $j = 1$ to dim **do**
      Ажурирање вредности A Једначином 3.65
      if $t = BS_k$ **then**
        Потрага за позицијом другог решења
        Ажурирање вредност BS_k Једначином 3.61
        Ажурирање простора претрага Једначином 3.63 и Једначином 3.64
        Дистрибуција свих решења $X = rand(N, dim) \times (ub - lb) + lb$
      end if
      if $A > 1$ **then**
        Почетак фазе експлорације
        Ажрурирање вредности W_1 и W_2 Једначином 3.54 и Једначином 3.56, респективно
        if $t \leq T$ **then**
          Прва фаза експлорације
          Ажурирање позиција потенцијалних решења Једначином 3.53
        else
          Друга фаза експлорације
          Ажурирање позиција потенцијалних решења Једначином 3.55
        end if
      else
        Почетак фазе експлоатације
        Ажурирање вредности W_3 Једначином 3.59
        if $t \leq T$ **then**
          Прва фаза експлоатације
          Ажурирање позиција потенцијалних решења Једначином 3.58
        else
          Друга фаза експлоатације
          Ажурирање позиција потенцијалних решења Једначином 3.60
        end if
      end if
    end for
  end for
  $t \leftarrow t + 1$
end while
Излаз: Најбоље решење $Best(X)$

3.10 *COLSHADE*

COLSHADE је алгоритам проистекао из *L-SHADE* алгоритма. Кључне промене чине механизам Леви летова и динамичка толеранција. Улога методе Леви лета се не примењује као код других алгоритама у експлорацији, већ одређују притисак избора над популацијом за проналажење погодног региона претраге као и одржавање различитости. Из тог разлога Леви летовима се уводи нови параметар мутације у овом алгоритму који се зове *levy/1/bin*. Овај механизам је најважнији за фазу експлорације док се за другу фазу примењује мутација тренутног према *pbest* решењу. Адаптивни механизам промене контролише употребу две врсте мутација у различитом капацитету и временским интервалима.

Велики број проблема из стварног живота припада категорији ограничене оптимизације, што се формално може представити за проблеме опште ограничене оптимизације (енгл. - *general constrained optimization problem*) Изразом 3.66.

минимизација $f(x), \quad x = (x_1, \ldots, x_D) \in \Omega$ подлеже $\begin{cases} g_i(x) \leq 0, & i = 1, \ldots, p, \\ h_j(x) = 0, & j = p+1, \ldots, m. \end{cases}$ (3.66)

Трансформисана једнакост ограничења је представљена Једначином 3.67.

$$|h_j(x)| - \epsilon \leq 0, \text{ for } j = p+1, \ldots, m, \tag{3.67}$$

из чега се може закључити да *COLSHADE* поседује стопу укрштања (енгл. - *crossover rate - CR*), фактор скалирања (енгл. - *scaling factor - SF*), мутациони оператор диференцијалне еволуције/тренутне до *pbest* мутације (енгл. - *diferential evolution/current-to-pbest - DE/cpb*), и линеарно опадајућу популацију.

Леви летови су уведени у овај алгоритам ради одржавања разноликости популације и да ублаже негативан ефекат који постиже селекција. Циљ ове методе јесте да побољша експлорацију. Примењује се насумични ход са дужином корака која је симулирана из дистрибуције са тешким репом (енгл. - *heavy tail distribution*). Леви летове карактерише велики број малих корака са спорадичним већим корацима.

Ограничења у *COLSHADE* алгоритму су представљена на другачији начин и ова метода се назива раздвајање ограничења и циља. По природи ово је похлепна стратегија, јер је квалитетнији узорак који је мање пута прекршио ограничења, а у случају пат позиције боље решење је оно које је пронашло бољу циљну вредност. Применом оваквог механизма могуће је искоренити јединке које нису погодне, али је неопходно оставити и неки проценат оваквих решења због разноликости и боље експлорације.

Унапређење диференцијалне еволуције јесте *L-SHADE* алгоритам. Мутациона стратегија *L-SHADE* алгоритма је представљена Једначином 3.68.

$$v_{i,g} = x_{i,G} + F_i \cdot (x_{\text{pbest},G} - x_{i,G}) + F_i \cdot (x_{r1,G} - x_{r2,G}), \tag{3.68}$$

где је $x_{pbest,G}$ насумично изабран од $p - best$ решења.

L-SHADE чува вредности M_cR и M_F за H уноса контролних параметара CR и F, респективно. Током иницијализације вредност меморије се подешава на 0.5. За време сваке генерације, контролни параметри CR_i и F_i су генерисани за свако могуће решење употребом насумичног индекса r_i у опсегу $[1, H]$ са следећим критеријумима:

Генерисање контролних параметара се врши по Једначини 3.69.

$$CR_i = \begin{cases} \text{rand}_n(M_{CR,r_i}, 0.1), & \text{if } M_{CR,r_i} > 0. \\ 0, & \text{у супротном,} \end{cases} \tag{3.69}$$

за фактор скалирања F описан Једначином 3.70.

$$F_i = \text{rand}_c(M_{F,r_i}, 0.1). \tag{3.70}$$

Сума ограничења која су прекршена се рачуна Једначином 3.71.

$$svc(x, \epsilon_G) = I(x) + E(x, \epsilon_G), \tag{3.71}$$

где је $\mathcal{I}(x)$ описано Једначином 3.72, а $\mathcal{E}(x, \epsilon_G)$ описано Једначином 3.73.

$$\mathcal{I}(x) = \sum_{i=1}^{p} \max(g_i(x), 0), \tag{3.72}$$

$$\mathcal{E}(x, \epsilon_G) = \sum_{j=p+1}^{m} \max(|h_j(x)| - \epsilon_{j,G}, 0). \tag{3.73}$$

Ажурирање толеранције се врши по Једначини 3.74.

$$\epsilon_{j,G+1} = \begin{cases} \epsilon_{j,G} \times \left(\frac{\epsilon_f}{\epsilon_{j,G}}\right)^{1/(FEs_\epsilon - FEs)}, & \text{ако су } p_G \text{ чланови популације } \epsilon\text{-погодни,} \\ \epsilon_{j,G}, & \text{у супротном.} \end{cases} \tag{3.74}$$

Бинарни турнир изводљивости се врши по Једначини 3.75.

$$x_{i,G+1} = \begin{cases} u_i, & svc(u_i; \epsilon_G) < svc(x_{i,G}; \epsilon_G), \\ u_i, & svc(u_i; \epsilon_G) = svc(x_{i,G}; \epsilon_G) = 0 \\ & \quad \wedge f(u_i) < f(x_{i,G}), \\ u_i, & svc(u_i; \epsilon_G) = svc(x_{i,G}; \epsilon_G) = 0 \\ & \quad \wedge svc(u_i; \epsilon_f) < svc(x_{i,G}; \epsilon_f), \\ x_{i,G}, & \text{у супротном.} \end{cases} \tag{3.75}$$

За руковање ограничењима, следећа насумична комбинација се користи по Једначини 3.76.

$$u_{i,j,G} = \begin{cases} (1 - r) \cdot U_j + r \cdot x_{i,j,G}, & \text{ако } u_{i,j,G} > U_j, \\ (1 - r) \cdot L_j + r \cdot x_{i,j,G}, & \text{ако } u_{i,j,G} < L_j. \end{cases} \tag{3.76}$$

где су L_j и U_j доња и горња граница променљиве x_j, респективно и r је случајни број r $U(0, 0.1)$.

Израчунавање једног унапређења се врши Једначином 3.77.

$$\Delta f_i = \begin{cases} svc(x_i; \epsilon) - svc(u_i; \epsilon), & \text{ако је погодност унапређена,} \\ f(x_i) - f(u_i), & \text{ако је оптималност унапређена.} \end{cases} \tag{3.77}$$

Лемер средња тежинска вредност се рачуна по Једначини 3.78.

$$L_2(S;w) = \frac{\sum_{s\in S} w_s \cdot s^2}{\sum_{s\in S} w_s \cdot s}, \tag{3.78}$$

где је w_s израчунато употребом Једначине 3.79.

$$w_s = \frac{\Delta f_s}{\sum_j \Delta f_j}, \tag{3.79}$$

Циклична меморија M се рачуна по Једначини 3.80.

$$M_{k,G+1} = \begin{cases} L_2(S;w), & \text{ако } |S| > 0, \\ M_{k,G+1}, & \text{у супротном.} \end{cases} \tag{3.80}$$

Стопа селекције мутационие стратегије се рачуна Једначином 3.81.

$$q_{G+1} = \begin{cases} q_G, & \text{ако } |\Delta f_L| = |\Delta f| = 0, \\ \mu \cdot q_G + (1-\mu)\cdot \hat{q}_G, & \text{у супротном,} \end{cases} \tag{3.81}$$

где је $\hat{q}_G$ описано Једначином 3.82.

$$\hat{q}_G = \frac{\sum_{i=1}^{|\Delta f_L|} \Delta f_{L,i}}{\sum_{i=1}^{|\Delta f_L|} \Delta f_{L,i} + \sum_{j=1}^{|\Delta f|} \Delta f_j}. \tag{3.82}$$

Псеудокод механизма Леви летова је описан Алгоритмом 10, а *COLSHADE* је приказан Алгоритмом 11.

Алгоритам 10 Псеудокод *levy/1/bin mutation.*

$F_{\text{crit}} \leftarrow \sqrt{(1 - CR_i/2)/N_G}$
$F_i \leftarrow \max(F_{\text{crit}}, F_i)$
$j_{\text{rand}} \leftarrow \text{RandInteger}(1, D)$
Изабери x_{pbest} насумично из P_G
for $j = 1$ до D **do**
  if $\text{rand}(0,1) \leq CR$ или $j = j_{\text{rand}}$ **then**
    $F_{\text{levy}} \leftarrow F_i \cdot S(\alpha, \beta, \gamma, \delta)$
    $u_j \leftarrow x_j + F_{\text{levy}} \cdot (x_{\text{pbest},j} - x_j)$
  else
    $u_j \leftarrow x_j$
  end if
end for

Алгоритам 11 Псеудокод *COLSHADE*.

$N_{\text{init}} \leftarrow \text{round}(D \times rN_{\text{init}})$
$N_0 \leftarrow N_{\text{init}}$, $N_{\min} \leftarrow 4$, $q_0 \leftarrow 0.5$
Иницијализација меморије $M_{CR}, M_F, M_{CRL}, M_{FL}$ на вредност 0.5
Иницијализација популације $P_0 = \{x_1, x_2, \ldots, x_{N_0}\}$
Подешавање почетне толеранције тако да су све једнакости ограничења могуће: $\epsilon_{0,j} = \max |h_j(x_i)|$
while Док услов искључења није постигнут **do**
  $SCR, SF, SCRL, SFL \leftarrow \emptyset$
  $\Delta f, \Delta fL \leftarrow \emptyset$
  for $i = 1$ to N_G **do**
    $l \leftarrow \text{rand}(0, 1)$
    if $l \leq q_G$ **then**
      $CR_i, F_i \leftarrow$ Генерисање параметара(M_{CRL}, M_{FL})
      $u_{i,G} \leftarrow$ *current-to-pbest/1/bin*$(x_i, CR_i, F_i, P_G, pbest)$
    else
      $CR_i, F_i \leftarrow$ Генерисање параметара(M_{CR}, M_F)
      $u_{i,G} \leftarrow$ *levy/1/bin*$(x_i, CR_i, F_i, P_G, pbest)$
    end if
    if $u_{i,G}$ унапређује $x_{i,G}$ **then**
      $x_{i,G+1} \leftarrow u_{i,G}$
      Копирање $x_{i,G}$ у архиву A
      if $l \leq q_G$ **then**
        Ажурирање $SCRL$, SFL, ΔfL
      else
        Ажурирање SCR, SF, Δf
      end if
    end if
  end for
  Ажурирање меморија M_{CR}, M_F и M_{CRL}, M_{FL}
  Ажурирање вероватноћа q_{G+1}
  Ажурирање толеранције ϵ_{G+1}
  Промена величине популације на основу N_{G+1}
  Промена величине архиве по потреби
end while

Глава 4

Остале методе

Узимајући у обзир да је тема рада уско везана за метахеуристичке алгоритме и то прецизније алгоритме *SI* ова глава докторске тезе је посвећена свим осталим типовима метода вештачке интелигенције која су примењена у овом раду. Другим речима, проблеми из стварног света чија су решења предложена у овом раду захтевају синергију са другим облицима вештачке интелигенције као што су на пример *LSTM* мреже које су погодне за решавање проблема чији су подаци облика временске серије. Поред наведеног типа мрежа описани су још и типови као што су *CNN*, бустинг методе (енгл. - *boosting*) и *NLP*. Ова глава научног рада закључује тему примене најразличитијих врста вештачке интелигенције заједно са великим бројем различитих врста метахеуристичких решења описаних у претходној глави.

4.1 Обрада природног језика

Значај области *NLP* је од својих теоријских темеља из 1940. била јасна. Да би рачунарство даље напредовало потребно је да овлада језиком који је основно средство комуникације за људе. Ова врста језика се назива природним, јер је развојем рачунарства настала нова врста језика - рачунарски. Дакле, рачунари такође поштују скупове правила у интерној и екстерној комуникацији. Узимајући у обзир јасне циљеве области вештачке интелигенције која је настајала у истом времену када и *NLP* област, главни циљ је *AGI* [41].

Дакле, може се рећи да развој *NLP* представља квалитет комуникације са машинама, односно рачунарима. Важност области је вишеструка за стварни живот јер ове методе представљају медијум који побољшава пренос информација између природног и рачунарског језика. То значи да је могуће прецизније извући значење из текста, као и лакши проналазак језичких образаца који нису видљиви голим оком и захтевају напрезање при анализи без помоћи других техника. Поред лингвистичког значаја, допринос у комуникацији може довести до мањих губитака информација што резултује квалитетнијим моделима за предикцију датог проблема. Неке од очигледних примена које могу имати бенефите од примене ових техника укључују: груписање текстуалних датотека [42], филтрирање спам електронске поште [43, 44], анализа сентимента [45], унапређење корисничког искуства [46], предикције берзе на основу вести [47] и класификација рецензија [48].

4.1.1 Двосмерни енкодер из репрезентација трансформатора

Представљен 2018, двосмерни енкодер из репрезентација трансформатора (енгл. - *bidirectional encoder representations from transformers - BERT*) је заснован на механизму пажње који служи за утврђивање значења текста [49]. Методу је развио истраживачки тим из компаније *Google* и од тада се примењује за разне проблеме из обраде природног језика [50]. Ова метода може бити модификована за посебне намене као што су сумаризација текста или закључивање значења. Механизам пажње омогућава трансформаторима промену фокуса по сегментима уноса. Употребом *BERT* методе разумевање реченица је унапређено јер захваљујући овој техници могуће је утврдити значење сваке речи. Додатно, могуће је овај процес примењивати паралелно што свеукупно подиже ефикаснсот процеса. За претходно описано је заслужан двосмерни механизам који анализира речи које долазе пре речи за коју се врши предикција, као и за оне после ње.

Тренирање ове методе се врши употребом модела језичке маске (енгл. - *masked language model - MLM*). Циљ овог типа тренинга јесте да сакрије једну реч коју је потребно предвидети само на основу контекста. На овај начин модел који се тренира може научити структуру и шаблоне датог језика. Тренирање *BERT* модела на огромним скуповима података може служити као припрема за примене са посебном наменом захваљујући способности преношења научених принципа (енгл. - *transfer learning*).

4.1.2 Техника учесталости појма - инверзне учесталости документа

Техника учесталости појма - инверзне учесталости документа (енгл. - *term frequency inverse document frequency - TF-IDF*) је заснована на статистичкој анализи текста [51]. Као што име технике сугерише, састоји се из два дела. Учесталост појма се одређује као однос појављивања појма и укупног броја појмова у документу. Описано је приказано Једначином 4.1.

$$\mathrm{TF}(t,d) = \frac{\text{Number of times term } t \text{ appears in document } d}{\text{Total number of terms in document } d}. \tag{4.1}$$

Други део се односи на инверзну учесталост документа, односно учесталост појављивања појма у свим документима који се анализирају. Улога овог механизма јесте да смањи важност речи које се често понављају у различитим документима јер се сматра да су то најчешће помоћне речи које не носе информације. Инверзна учесталост документа се рачуна по следећој Једначини 4.2.

$$\mathrm{IDF}(t,D) = \log\left(\frac{N}{|\{d \in D : t \in d\}|}\right). \tag{4.2}$$

Комбиновањем учесталости појма и инверзне учесталости документа израчунава се важност појма у документу у односу на цео скуп података по Једначини 4.3.

$$\text{TF-IDF}(t,d,D) = \mathrm{TF}(t,d) \times \mathrm{IDF}(t,D). \tag{4.3}$$

4.2 Конволуцијске неуронске мреже

Конволуцијске неуронске мреже, *CNN* су препознатљиве у области дубоког учења због високих перформанси [52]. Вишеслојни визуелни кортекс животиња је служио као главна инспирација за ову методу. Информација пролази кроз слојеве мреже где је излаз претходног слоја улаз у наредни слој. На овај начин информације се филтрирају и обрађују. Како се информација креће кроз слојеве њена комплексност се смањује а способност модела да опажа финије детаље расте. Архитектура *CNN* се састоји из конволуционог слоја, слоја сажимања и потпуно повезаног слоја. Најчешће коришћени филтери су 3×3, 5×5, и 7×7.

Да би максималне перформансе биле постигнуте потребно је оптимизовати хиперпараметре *CNN*. Параметри који су најчешће оптимизовани засновано на утицају на перформансе су број кернела и њихова величина, степен обучавања, величина серије, број слојева по типовима, регуларизација тежине, функција активације и стопа одустајања.

Функција конволуције даје улазни вектор описан Једначином 4.4.

$$z_{i,j,k}^{[l]} = w_k^{[l]} x_{i,j}^{[l]} + b_k^{[l]}, \tag{4.4}$$

где је излаз k-тог обележја на позицији i, j, слој l је представљен као $z_{i,j,k}^{[l]}$, улази на i, j су x, филтери за приказ w и пристрасност је b.

Активациона функција прати конволуциону што је описано Једначином 4.5.

$$g_{i,j,k}^{[l]} = g(z_{i,j,k}^{[l]}), \tag{4.5}$$

где је нелинеарна функција експлоатације излаза представљена као $g(\cdot)$.

Након активационе функције, слојеви сажимања описани Једначином 4.6 обрађују улаз смањивањем резолуције. Различити слојеви сажимања могу бити употребљени, а неки од најпопуларнијих су просечно и максимално сажимање.

$$y_{i,j,k}^{[l]} = pooling(g_{i,j,k}^{[l]}). \tag{4.6}$$

где y представља резултат сажимања.

На крају, класификација се врши помоћу потпуно повезаних слојева. Најчешће коришћене имплементације ових слојева су *softmax* за потребе вишекласне класификације, и сигмоидна функција за потребе бинарне класификације уз методу градијентног спуста.

Тежине и пристрасности се подешавају у свакој итерацији. Циљ је да се минимизује функција губитака описана Једанчином 4.7.

$$H(p,q) = -\sum_x p(x) ln(q(x)), \tag{4.7}$$

где дискретна променљива x има две дистрибуције, p и q.

4.3 Дуга краткорочна меморија

Разлог иза развоја рекурентних неуронски мрежа (енгл. - *recurrent neural network* - *RNN*) јесте решавање проблема предвиђања временске серије [53]. Могуће је упоредити регуларни вишеслојни перцептрон (енгл. - *multilayer perceptron* - *MLP*) са

разликом што *RNN* поседује и скривене конекције које су омогућене са задршком. Последица ове разлике јесте могућност *RNN* да манипулише релацијама између временски удаљених података што је комплексан задатак.

Поред унапређених способности код *RNN* ове мреже нису без мана. Проблем при тренингу представља нестајање, као и експлозија градијента. Међутим, већ је предложено решење овог проблема а то су *LSTM* мреже. Међутим, узимајући у обзир *NFL* теорему као и већу комплексност оптимизације *LSTM*, *RNN* може да постигне боље перформансе за проблеме са мањом комплексношћу података.

Предност *RNN* мрежа је у томе што не захтевају унапред познату дужину улазних вектора, како и излазних. Другим речима, не само да је могуће радити са различитим секвенцама улаза, већ је могуће и генерисање излаза. Задржавајући сакривено стање, *RNN* обрађује податке из секвенце.

Још напреднија, али и компликованија верзија је *LSTM* која задржава ове меморијске функције. Проблеми временске серије захтевају могућност управљања дугорочним конекцијама између података. Меморијске ћелије скривених слојева су замењене и примењују се три капије (енгл. - *gates*) и то улазна (енгл. - *input*), излазна (енгл. - *output*) и капија заборава (енгл. - *forget*).

Подаци пролазе кроз капију заборава f_t која је заслужна за одлучивање о заборављању података и описана Једначином 4.8.

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f), \tag{4.8}$$

где је распон f_c капије $[0, 1]$, сигмоидна функција је приказана као σ, док су матрице променљивих приказане као W_f и U_f, а b_f је вектор пристрасности.

Следећа на реду је улазна капија која одлучује који подаци ће бити сачувани, приказана у Једначини 4.9.

$$i_t = \sigma(W_i X_t + U_i h_{t-1} + b_i), \tag{4.9}$$

где је $[0, 1]$ распон i_t, док су обучиви параметри b_i W_i, и U_i. Потенцијалне векторе за ажурирање C_t је могуће срачунати Једначином 4.10.

$$C_t = tanh(W_c x_t + U_c h_{t-1} + b_c), \tag{4.10}$$

где су обучиви параметри b_c, W_c U_c. Наставак овог процеса је описан у Једначини 4.11.

$$C_t = F_t \odot C_{t-1} + i_t \odot C_t, \tag{4.11}$$

где $\odot$ представља Хадамардов производ. Подаци спремни за заборав су обележени као C_{t-1}. Подаци за чување су $f_t \odot C_t$, док се нови подаци $i_t \odot C_t$ чувају у C_t. Скривено стање h_t вредности o_t излаз из последње капије се рачуна сигма функцијом описаном Једначином 4.12.

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o), \tag{4.12}$$

где је h_t описано Једначином 4.13.

$$h_t = o_t \odot tanh(C_t), \tag{4.13}$$

где је o_t дефинисано распоном $[0, 1]$ са b_o W_o и U_o обучивим параметрима. Излаз h_t је представљен као o_t и $tanh$ вредност производа C_t.

4.4 Адаптивни бустинг

AdaBoost групише више слабијих алгоритама са циљем бољих перформанси. Алгоритам су развили Јоав Фројнд (енгл. - *Yoav Freund*) и Роберт Шапир (енгл. - *Robert Schapire*) [54]. Алгоритми који се сматрају слабим извршавају класификацију мало боље него насумично погађање. Кроз итерације се примењује још слабих класификатора као и балансирање тежина класификатора изведених из тачности. Приликом грешака у класификацији, тежине се смањују, док се приликом успешне класификације повећавају,

Грешка слабог класификатора се рачуна по Једначини 4.14.

$$\epsilon_t = \frac{\sum_{i=1}^{N} w_{i,t} \cdot \mathrm{I}(h_t(x_i) \neq y_i)}{\sum_{i=1}^{N} w_{i,t}}, \tag{4.14}$$

где је тежина грешке у t-тој итерацији ϵ_t, број узорака за тренирање N, тежина i-тог тренинг узорка за време t-те итерације $w_{i,t}$, предвиђена ознака $h_t(x_i)$, а стварна ознака је y_i и функција $\mathrm{I}(\cdot)$ има резултат 0 за погрешне случајева, а 1 за тачне.

Након додељивања тежина отпочиње процес модификације тежина за нове класификаторе. Да би модел био тачан у предикцијама потребно је користити велики број алгоритама. Комбинација подмодела и њихових резултата представља линеарни модел. Тежине се израчунавају ансамблно по Једначини 4.15.

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right), \tag{4.15}$$

где се α_t мења за сваки слаби алгоритам и представља тежину у коначном моделу. Тежине се ажурирају по Једначини 4.16.

$$w_{i,t+1} = w_{i,t} \cdot \exp\left(-\alpha_t \cdot y_i \cdot h_t(x_i)\right), \tag{4.16}$$

где y_i означава истиниту ознаку i-те инстанце, $h_t(x_i)$ представља предикцију слабог алгоритма i-те инстанце у t-тој рунди и $w_{i,t}$ представља тежину у i-тој инстанци у t-тој рунди.

Предност овог ансамбл алгоритма је смањивање пристрасности преко учења из претходних итерација, док техника ансамбла смањује варијансу и спречава претренирање (енгл. - *overfitting*). Могуће је произвести робусне моделе уз помоћ ове технике, али је осетљива на шум и изузетке.

4.5 Екстремни градијентни бустинг

XGBoost је алгоритам високих перформанси [55]. Основа алгоритма је ансамбл учење и експлоатација слабијих модела. Оптимизација са регуларизацијом и градијентим бустингом омогућава знатне напретке у перформансама. Мотивација за настанак оваквог решења је управљање комплексним односима између улаза и циљне променљиве уз помоћ претходно опажених шаблона.

Циљна функција овог алгоритма комбинује функцију губитка са регуларизацијом што је описано Једначином 4.17.

$$\mathrm{obj}(\Theta) \;=\; L(\theta) \;+\; \Omega(\Theta), \tag{4.17}$$

где Θ приказује скуп хиперпараметара, $L(\Theta)$ је функција губитка, а $\Omega(\Theta)$ представља регуларизацију за управљење комплексношћу модела.

Средња квадратна грешка (енгл. - *mean square error - MSE*) је употребљена за функцију губитка описана Једначином 4.18.

$$\mathrm{L}(\Theta) \;=\; \sum_i (y_i \;-\; \hat{y_i})^2, \tag{4.18}$$

где је $\hat{y_i}$ предвиђена вредност циљне променљиве за сваку итерацију i, а y_i предвиђена вредност.

Разликовање стварних и предвиђених вредности се може описати Једначином 4.19.

$$L(\Theta) \;=\; \sum_i [\; y_i \ln\,(1 + e^{-\hat{y}_i}) \;+\; (1 - y_i) \ln\,(1 + e^{\hat{y}_i})]. \tag{4.19}$$

Глава 5

Модификације алгоритама

Модификације *PSO* алгоритма су главни проблем којим се бави ова теза. Предложене су две нове верзије овог алгоритма са различитим унапређењима у односу на проблем којим се баве. Верзија за проблеме поморског саобраћаја, који чине проблем класификације типа пловила и проблем регресије путање пловила и верзија за проблем класификације софтверске грешке.

5.1 Модификације алгоритма ројева честица према проблемима поморског саобраћаја

Проблеми поморског саобраћаја представљају посебан изазов због саме њихове природе. Поморски саобраћај прати велики број непредвидих фактора који утичу на његову динамику. Модерни уређаји за праћење и навигацију су далеко од савршеног што често има за резултат да пловило није видљиво на уређајима за праћење на копну. Ови временски прозори остављају простор за малверзације, што од саме посаде, што од пиратских бродова. Предвиђање типа пловила методама вештачке интелигенције може унапредити безбедност руте, као што може и унапредити предицкију саме руте што је подржано предстојећим резултатима.

5.1.1 Бустована оптимизација ројевима честица

Основна верзија *PSO* алгоритма се као таква сматра за робусни алгоритам. Резултати референтних тестова (енгл. - *benchmark tests*) указују на недостатке које је могуће унапредити у односу на основу верзију *PSO* [56]. Основно унапређење се тиче способности експлорације као и брзине конвергенције. Могућа су унапређења и на пољу избегавања локалних оптимума. Два нова механизма су предложена за унапређење овог алгоритма за оптимизацију проблема поморског саобраћаја.

5.1.1.1 Елитна хаотична стратегија обучавања

Прва уведена измена је додавање елитне хаотичне стратегије обучавања (енгл. - *chaotic elite learning technique* - *CEL*), која може помоћи алгоритму да избегне преурањену конвергенцију у погрешне делове простора за претрагу, чинећи тако алгоритам ефикаснијим и способнијим да открије оптимална решења [57]. Део унапређења који утиче на индивидуалног агента у популацији се односи на хаотичну секвенцу које се

користи за производњу нових решења у околини. Због овога је алгоритам способан да лакше избегне локалне оптимуме, али је и разноликост популације такође боља.

Логистичка мапа преко које се уводи хаотично понашање је описана Једначином 5.1.

$$c_{i+1} = 4 \times c_i \times (1 - c_i), \tag{5.1}$$

где је број итерација обележен са i, хаотична вредност тренутне итерације i са c_i, а почетна вредност c_0 се насумично генерише у опсегу $[0, 1]$.

Ажурирање вредности најбољег решења P_i се рачуна по Једначини 5.2.

$$P'_{i,j} = P_{i,j} + rand \times (2 \times c_i - 1), \tag{5.2}$$

где $P'_{i,j}$ представља j компоненту најбоље позиције која се ажурира.

5.1.1.2 Леви летови

Насумични ход се показао као недовољно адекватна техника за унапређење процедуре ажурирања позиција, те се може применити механизам Леви летова [58, 59]. Идеја иза овог приступа јесте да се свакој јединци додели способност да прави знатно веће скокове али повремено. Последица овога понашања је излажење из субоптималних региона. Додатни механизам је имплементиран да смањује дистанцу скока при каснијим итерацијама што је описано Једначином 5.3 и Једначином 5.4.

$$X'_{i,j} = X_{i,j}^{best} \times L \times e, \tag{5.3}$$

$$e = \frac{\overrightarrow{a}}{2}, \tag{5.4}$$

где $X'_{i,j}$ представља j компоненту за итерацију i која се ажурира, док $X_{i,j}^{best}$ приказује усмерење ка оптималном решењу за индивидуу i. Фактор скалирања је уведен као симбол e, који се смањује кроз рунде, а $\overrightarrow{a}$ се добија применом Једначине 5.5.

$$\overrightarrow{a} = 1 + cos(\pi \times \frac{t}{T}), \tag{5.5}$$

где је тренутна рунда t, а максимални њихов број је T. Дистрибуција Леви летова је представљена следећом Једначином 5.6.

$$L = s \times \frac{u \times \phi}{|\nu|^{\frac{1}{\tau}}}, \tag{5.6}$$

где τ приказује Леви индекс, s је константа која има вредност 0.01 и служи да ограничи дужину скокова, параметри u и ν су насумичне вредности из опсега $[0, 1]$. Параметар ϕ се израчунава по Једначини 5.7.

$$\phi = \left(\frac{\Gamma(1+\tau) \times sin(\frac{\pi \times \tau}{2})}{\Gamma(\frac{1+\tau}{2}) \times \tau \times 2^{\frac{\tau-1}{2}}} \right)^{\frac{1}{\tau}}, \tag{5.7}$$

где је Γ приказана Једначином 5.8.

$$\Gamma(1+\tau) = \int_0^\infty x^\tau e^{-x}\, dx. \tag{5.8}$$

5.1.2 Псеудокод модификованог алгоритма

Циљ измена на основни алгоритам је према побољшању експлоатације и зато је и алгоритам назван *BPSO*. Прва измена, *CEL*, је примењена у свакој итерацији на тренутно пронађено најбоље решење. Леви летови су зато примењени као стандардан механизам претраге. Промена позиције решења се врши или преко стандардне претраге *PSO* алгоритма или путем Леви летова. Одлуку о примени претраге доноси параметар ϕ, униформно расподељен у интервалу $[0, 1]$ и генерисан за свако решење у свакој итерацији. Наведене промене имају задатак да донесу баланс између фаза експлорације и експлоатације.

Приказан је упрошћени псеудокод *BPSO* Алгоритмом 12, где је максимални број итерација T узет као услов изласка из претраге .

Алгоритам 12 Псеудокод *BPSO* алгоритма.

Иницијализација статичких и динамичких контролних параметара
Иницијализација низа популације насумичних позиција и брзина димензија D у простору претраге
repeat
  Евалуације функције погодности за сваку честицу D променљивих
  Поређење исхода са $pbest_i$.
  if Тренутна вредност боља од $pbest_i$ **then**
    Тренутна вредност постаје $pbest_i$
    $\overrightarrow{p_i}$ добија вредност тренутне позиције $\overrightarrow{x_i}$ у D-димензионалном простору
  end if
  Променљива g добија вредност индекса тренутно најбољег решења
  if ϕ<=0.5 **then**
    Честице мењају позицију на основу Једначине 3.2
  else
    Честице мењају позицију на основу Једначине 5.3
  end if
  Честице мењају позицију хаотичном Једначином 5.2
  Ажурирање вредности контролних параметара
until Критеријум је задовољен

Предложене модификације не уводе додатну комплексност по питању рачунарске сложености у односу на основни алгоритам на основу евалуације функције погодности (енгл. - *fitness function evaluations - FFE*). Дакле комплексност основног *PSO* и *BPSO* се може представити као $O(N) = N + N \cdot T$, где је број решења у популацији N.

5.2 Модификације алгоритма оптимизације ројевима честица према проблемима детекције грешака у коду

Детекција софтверских грешака постаје све важнији проблем јер софтвер можемо пронаћи готово у свему. Паметни уређаји су све чешћи феномен, где више није

јасно да ли је стварно потребно да ти уређаји буду опамећени. Више кода значи и више прилике за грешку, указујући на то да је потребно пронаћи адекватан приступ како би се ова статистика обуздала. Посебну пажњу је потребно посветити када је потребно ослонити се на софтвер у ситуацијама у којима постоји могућност губитка људског живота. Измене *PSO* алгоритма према успешнијем решавању овог критичног проблема за даљи развој цивилизације су приказане у даљем тексту.

5.2.1 Мулти-популацијска оптимизација ројевима честица

Осврћући се на напретке у области оптимизације, могуће је на још један начин унапредити перформансе *PSO* алгоритма. Недостаци основне верзије овог алгоритма се огледају у фази експлорације, те је унапређење могуће у овом сегметну. Предложене промене укључују хибридизацију са *GA* и механизмима инспирисаним еволуцијом. Из овог разлога је предложено решење названо мулти-популацијски *PSO* (енгл. - *multi-population PSO - MPPSO*). Први корак предложеног алгоритма чини генерисање популације где су и прве измене приметне. Прва половина генерисаних агената се добија помоћу Једначине 5.9.

$$X_{i,j} = lb_j + \psi \cdot (ub_j - lb_j), \tag{5.9}$$

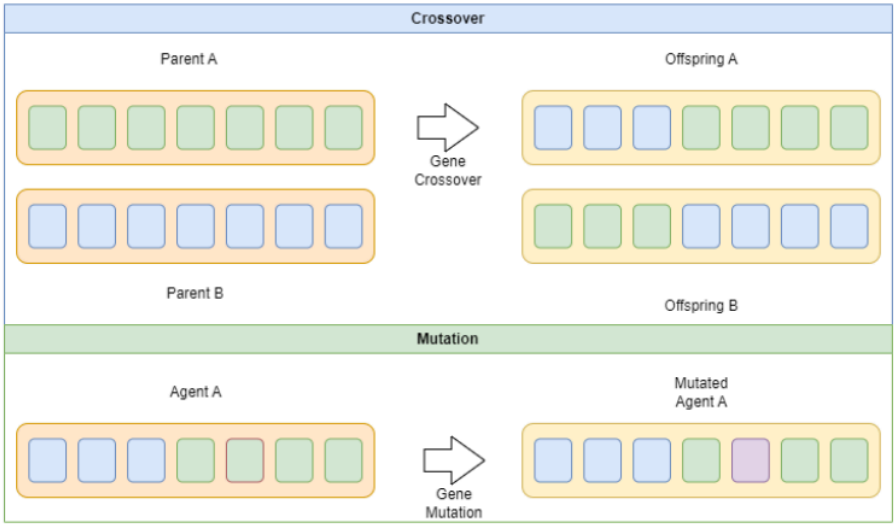
где је $X_{i,j}$ j-ти фактор додељен индивидуи i, lb_j и ub_j представљају границе j, ψ је елемент који уводи насумичност из униформне расподеле између $[0, 1]$.

Остатак популације се генерише помоћу квази-рефлексивног учења (енгл. - *quasi-reflection-based learning - QRL*) по Једначини 5.10.

$$X_j^{qr} = \text{rnd}\left(\frac{lb_j + ub_j}{2}, x_j\right), \tag{5.10}$$

где је rnd насумична вредност из интервала $\left[\frac{lb_j + ub_j}{2}, x_j\right]$. Диверзификација трпи унапређења применом *QRL* на стратегију иницијализације.

Популације генерисане са две различите описане технике остају подељене за време оптимизације. Механизми генетског укрштања и мутације повезују ове популације. Укрштањем се бирају најбољи параметри из оба агента која су предмет ове операције, што даје за резултат супериорнију популацију. Додатна насумичност је уведена путем методе мутације која параметре у постојећим члановима популације насумично мења у одређеном интервалу. Ова два процеса су описана Сликом 5.1.



Слика 5.1: Механизми укрштања и мутације.

Модификовани алгоритам доприноси балансу фаза експлорације и експлоатације модификовањем чланова популације који учествују у укрштању. Током почетних $\frac{1}{2}$ итерација, два насумична агента се за сваку потпопулацију бирају и рекомбинују. Затим се примењује мутација са вероватноћом мутације mp од 0.1. Слично, укрштање између агената је регулисано параметром вероватноће укрштања cp постављеним на емпиријски утврђену вредност од 0.1 за ово истраживање. Друга половина итерација се разликује у томе што се овај процес извршава за најбоља два агента из популације. Стога се експлорација подстиче у раним фазама оптимизације, а експлоатација је побољшана у каснијим фазама.

Псеудокод предложених измена је приказан Алгоритмом 13.

Алгоритам 13 Псеудокод *MPPSO*.

Иницијализација $\frac{1}{2}$ популације честица P низом са насумичним позицијама и брзинама у D променљивим
Примена *QRL* за генерисање остатка популације
Подела популације на две једнаке целине
repeat
  Евалуација помоћу одабране функције погодности сваке честице у D променљивим
  Поређење исхода са $pbest_i$
  if Тренутна вредност боља од $pbest_i$ **then**
    Подеси вредност $pbest_i$ на тренутну вредност
    Подеси $\overrightarrow{p_i}$ на тренутну локацију $\overrightarrow{x_i}$ у D-димензионом простору
  end if
  Додела индекса најбољег решења до сада променљивој g
  Промена позиције честица путем Једначине 3.2
  for all Агенти у популацији P **do**
    Генерисање насумичне вредности R у распону $[0, 1]$
    if $R > cp$ **then**
      if t $< \frac{T}{2}$ **then**
        Примена укрштања на насумичне агенте из обе популације
      else
        Примена укрштања на најбоље агенте из обе популације
      end if
    end if
  end for
until Критеријум задовољен
return Најбољи агент као решење

5.2.2 Комплексност предложене мулти-популацијске оптимизацијеројевима честица

Када се анализира сложеност метахеуристичких метода, уобичајена пракса је да се она изрази у односу на *FFE* по извршавању алгоритма. *FFE* је препозната као најкомплекснија операција током извршавања метахеуристике, као што је приказано у бројним публикацијама [60, 61, 62, 63]. Ово истраживање прати устаљену праксу и користи *FFE* за процену сложености *MPPSO*.

Као што је претходно детаљно описано, предложени нови *MPPSO* уводи две модификације основног *PSO*. Међутим, пошто се нови појединци не процењују након стварања (њихова евалуација се одлаже на следећи круг), сложеност новог *MPPSO* остаје иста као и основни *PSO*. Рачунарска сложеност у велико O нотацији (енгл. - *big O notation*) је приказана Једначином 5.11.

$$O(N) = N + T \cdot N, \tag{5.11}$$

где је N број јединки у популацији, док T приказује максимални број итерација за време једног покретања.

Глава 6

Резултати

Предстојећа глава представља резултате оба предложена алгоритма. Прво су приказане све метрике које осликавају перформансе оба предложена алгоритма. Затим следе описи скупа података и подешавања специфичних за прву групу експеримената, праћени истим за другу групу. Најбољи резултати су дуж главе приказани подебљаним фонтом.

6.1 Примењене метрике за компаративну анализу

Два скупа метрика за евалуацију су примењена са циљем успостављања објективног критеријума за поређење перформанси оптимизационих алгоритама. За експерименте класификације су стандардне метрике примењене и то тачност, прецизност, одзив и $F1$ резултат. Додатно је примењен и Коенов Капа параметар (енгл. - *Cohen's kappa parameter*) чији циљ је да веродостојније представи перформансе на скуповима података где подаци нису избалансирани.

Тачност је описана Једначином 6.1.

$$Accuracy = \frac{TruePositives + TrueNegatives}{TruePositives + TrueNegatives + FalsePositives + FalseNegatives}. \tag{6.1}$$

Одзив је описан Једначином 6.2.

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}. \tag{6.2}$$

Презицност је описана Једначином 6.3.

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}. \tag{6.3}$$

$F1$ резултат је описан Једначином 6.4.

$$F_1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}. \tag{6.4}$$

Коенов Капа параметар је описан Једначином 6.5.

$$Cohen'sKappa = \frac{ObservedAgreement - ExpectedAgreement}{1 - ExpectedAgreement}. \tag{6.5}$$

За експерименте предвиђања путање пловила регресионе метрике су примењене. Оне укључују *MSE*, средњу апсолутну грешку (енгл. - *mean absolute error - MAE*), корен средње квадратне грешке (енгл. - *root mean squared error - RMSE*) и коефицијент одлучности R^2 (енгл. - *coefficient of determination*). Поред стандардних метрика примењени су и индекс усклађености (енгл. - *index of alignment - IoA*) и грешка Еуклидског растојања (енгл. - *Euclidean distance error - EDE*) која служи да одреди удаљеност предвиђене и стварне локације пловила.

MAE је описан Једначином 6.6.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|. \tag{6.6}$$

MSE је описан Једначином 6.7.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2. \tag{6.7}$$

RMSE је описан Једначином 6.8.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}. \tag{6.8}$$

R^2 је описано Једначином 6.9.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}. \tag{6.9}$$

IoA је описан Једначином 6.10.

$$IoA = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(|y_i - \bar{y}| + |x_i - \hat{y}_i|)^2}, \tag{6.10}$$

где y_i представља забележене вредности, $\hat{y}_i$ предвиђене вредности, $\bar{y}$ средњу вредност забележених вредности и n представља број забележених узорака.

EDE је описано Једначином 6.11.

$$EDE = \sqrt{(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2}, \tag{6.11}$$

где су координате пловила x_i и y_i, док су предвиђене вредности координата представљене као $\hat{x}_i$ и $\hat{y}_i$. Као усмерење процесу оптимизације је кориштена *MSE* метрика као циљна функција где је циљ минимизација. Средња вредност *EDE* метрике је кориштена за индикаторску функцију.

6.2 Експерименти предвиђања поморског саобраћаја

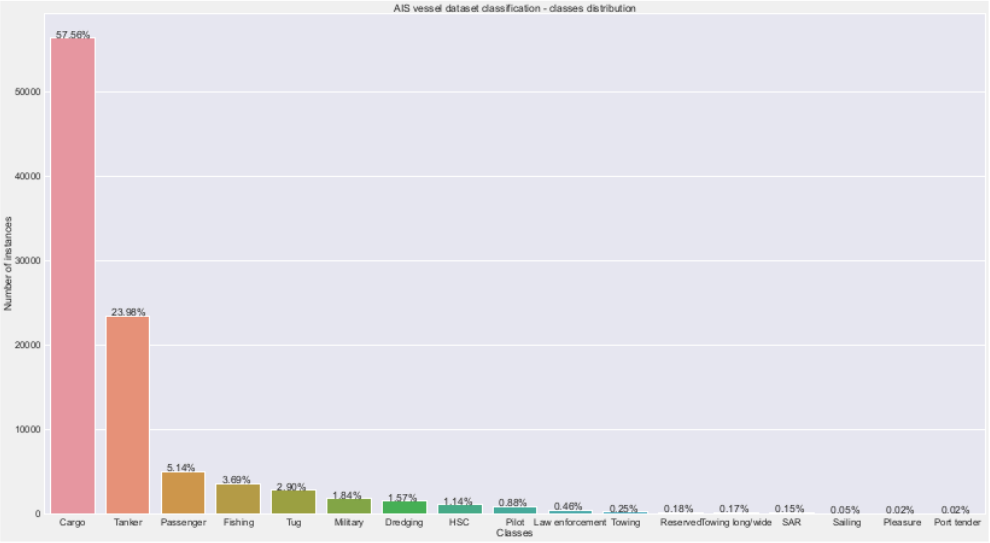
Експерименти у којима је тестиран *BPSO* алгоритам потврђују унапређење у односу на основну верзију *PSO* алгоритма. Алгоритам је тестиран као оптимизатор за две различите методе. Прво су извршени експерименти класификације пловила, где је циљ био оптимизовати хиперпараметре *XGBoost* модела задуженог за предикцију. Друга група експеримената је извршена са *LSTM NN* као предиктором путање пловила.

6.2.1 Примењени скупови података и предобрада података

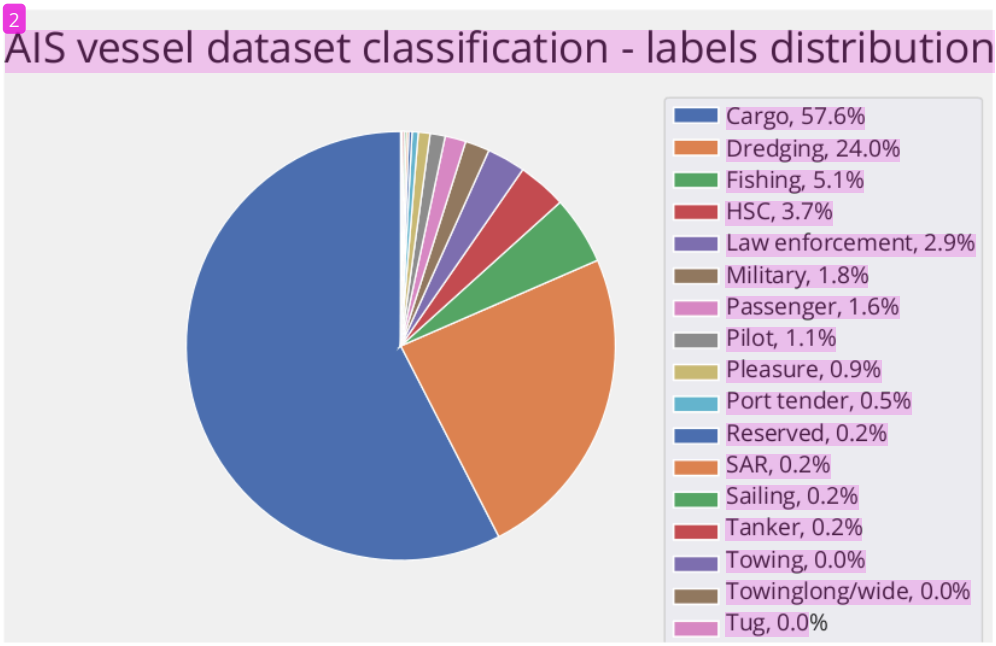
Скуп података који је употребљен у овој групи експеримената је јавно доступан и преузет са интернет адресе https://www.kaggle.com/datasets/eminserkanerdonmez/ais-dataset. Подаци су за овај скуп података прикупљени употребом *AIS*. Ови подаци су идентитет поморске мобилне услуге (енгл. - *maritime mobile service identity - MMSI*), статус навигације, брзина изнад дна, курс изнад дна, курс, ширина, дужина и газ. Наведена обележја су употребљена за идентификацију пловила.

Обележје које представља статус навигације је представљено као ниска (енгл. - *string*) и енкодовано је у целобројни тип (енгл. - *integer*) како би стабло одлучивања могло да ради са овим подацима. Међутим, овај приступ је давао лошије резултате него принцип енкодовања вруће јединице (енгл. - *one hot encoding*) и зато је одабран овај приступ.

Дистрибуција класа типа пловила је представљена Сликом 6.1 и Сликом 6.2.



Слика 6.1: Бар дијаграм дистрибуције класа.



Слика 6.2: Кружни дијаграм дистрибуције класа.

Скуп података за експерименте предвиђања путање пловила је сачињен од података преузетих из поморског катастра САД https://marinecadastre.gov/ais/. Обалска стража САД је прикупљала ове податке у интернационалним и водама САД помоћу уређаја за навигацију на пловилима. *AIS* подаци за 31.03.2022. су употребљени за овај експеримент. Због великог броја пловила одабран је један тип пловила за предвиђање путање.

Припрема скупа података за предвиђање путање је отпочета са уклањањем обележја *MMSI*, тип пловила, тип пријемника, навигациони статус, дужина и висина, јер нису релевантни за предвиђање путање. Преостала обележја чине географска ширина и дужина, брзина преко дна, курс изнад дна и газ. Следећа обрада се односи на интервале преноса информација који су у оригиналном скупу података неравномерни због природе пловидбе. Ови интервали су поново узорковани на интервале од 60 секунди, а недостајуће вредности су интерполиране путем методе полиномске интерполације. Последњу измену чини додатак две нове вредности извучене из газа и брзине изнад дна и то су брзина пловила у X и Y смеру. Помоћу нових обележја модел је боље учио зависности између газа и путање што је унапредило тачност предвиђања.

6.2.2 Подешавања експеримената

Експерименти су сачињени од поређења неколико метахеуристичких алгоритама високих перформанси са предложеним решењем. Сваки алгоритам је подешен да тражи скуп параметара који дају најбољи модел. Алгоритми који су тестирани укључују *PSO* [3], *GA* [33], *ABC* [35], *BA* [36], *WOA* [37], *HHO* [38] и *ChOA* [64].

За експерименте класификације употребљена је популација од 10 јединки и 10

итерација за оптимизацију хиперпараметера *XGBoost* модела. Параметри за оптимизацију су бирани на основу њиховог утицаја на перформансе модела. Границе у којима оптимизациона метахеуристика може да одређује вредност су $[0.1, 0.9]$ за *learning rate*, $[1, 10]$ за минималну тежину детета (енгл. - *min child weight*), $[0.01, 1]$ за подузорак (енгл. - *subsample*), $[0.01, 1]$ за узорак колона по дрвету (енгл. - *colsample by tree*), $[3, 10]$ за максималну дубину стабла (енгл. - *max depth*) и $[0, 0.8]$ за параметар гама. Скуп података чине подскупови од 70% укупних података за тренирање, 10% за валидацију и 20% за тестирање.

За предвиђање путање, моделу су прослеђени географска ширина и дужина, брзина изнад дна и курс изнад дна. Циљ ове групе експеримената је предвиђање географске ширине и дужине. Моделима *LSTM* је прослеђено 16 кашњења (енгл. - *lags*), односно узорака који су узети у обзир као улаз у мрежу, величина серије (енгл. - *batch size*) је 20 и примењена је *TimeSeriesGenerator Keras* библиотека за тренирање модела. Број кашњења је одлучен емпијерски као компромис између меморије и рачунске захтевности и тачности. Половина података је употребљена за тренирање, а друга половина за тестирање.

Метахеуристички алгоритми имају задатак да пронађу оптимални скуп параметара и оптималну архитектуру *LSTM* мреже. Још једном, параметри за оптимизацију су селектовани на основу њиховог утицаја на перформансе модела и то су степен обучавања у интервалу $[0.0001, 0.01]$, насумично искључивање (енгл. - *dropout*) у интервалу $[0.05, 0.2]$ и број тренинг епоха у интервалу $[30, 50]$. На крају је оптимизован број слојева мреже и то у интервалу $[1, 3]$. Број неурона у сваком слоју је такође оптимизован, али пошто зависи од броја примењених кашњења границе за овај параметар су $[lags, lags * 2]$. Није примењен валидациони сет, а подаци су груписани у односу 50/50 за тренинг и тест скупове.

6.2.3 Резултати, компаративна анализа, валидација и визуелизација

Ова глава описује исходе експеримената на основу претходно описаних подешавања. Прво су описани тестови класификације пловила, а затим исходи предвиђања путања пловила. Након тога су резултати валидирани статистичким тестовима и најбољи модели анализирани заједно са значајем обележја.

6.2.3.1 Закључци експеримената и компаративна анализа

Два различита проблема су решена са претходно описаним експериментима. Први проблем се односи на препознавање типа пловила, односно експерименти класификације. Други проблем се односи на предвиђање путање пловила, односно експерименте регресије. Обе групе експеримената су неопходне јер њихов комбиновани допринос је кључан у ситуацији која би се десила у стварном свету где би циљ био праћање и идентификација пловила. Сви подаци су прикупљени путем *AIS* уређаја где је за експерименте класификације циљ минимизација грешке, а за регресиону групу максимизација Коеновог Капа параметра.

6.2.3.2 Први експеримент: класификација морских пловила

Метрике за поређење перформанси током класификације су најбољи, најгори и средњи резултат, стандардна девијација и варијанса. Резултати минимизације грешке за експерименте класификације су приказани у Табели 6.1.

Табела 6.1: Укупни резултати циљне функције (грешка класификације) - Експеримент минимизације грешке класификације.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
XG-BPSO	**0.002789**	**0.003470**	**0.003100**	**0.003130**	**0.000202**	**4.09E-08**
XG-PSO	0.003096	0.005375	0.004010	0.003929	0.000739	5.47E-07
XG-GA	0.004354	0.015614	0.007037	0.005511	0.003554	1.26E-05
XG-ABC	0.004898	0.007586	0.005736	0.005477	0.000817	6.67E-07
XG-BA	0.003368	0.005171	0.004116	0.004184	0.000559	3.13E-07
XG-WOA	0.003198	0.004762	0.004065	0.004133	0.000463	2.14E-07
XG-HHO	0.003674	0.007076	0.004915	0.004286	0.001205	1.45E-06
XG-ChOA	0.003062	0.009899	0.004720	0.004133	0.002032	4.13E-06

Резултати Коенове Капе за експерименте класификације су приказани у Табели 6.2.

Табела 6.2: Укупни резултати индикаторске функције (Коенова Капа) - Експеримент минимизације грешке класификације.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
XG-BPSO	**0.995391**	**0.994267**	**0.994878**	**0.994829**	**0.000335**	**1.12E-07**
XG-PSO	0.994886	0.991114	0.993374	0.993507	0.001223	1.50E-06
XG-GA	0.992803	0.974130	0.988359	0.990888	0.005892	3.47E-05
XG-ABC	0.991899	0.987456	0.990516	0.990946	0.001351	1.83E-06
XG-BA	0.994435	0.991452	0.993197	0.993085	0.000925	8.56E-07
XG-WOA	0.994717	0.992126	0.993282	0.993169	0.000766	5.87E-07
XG-HHO	0.993928	0.988296	0.991873	0.992917	0.001995	3.98E-06
XG-ChOA	0.994942	0.983617	0.992197	0.993169	0.003366	1.13E-05

За оба случаја је предложено решење доминирало у свим категоријама. Детаљније информације као прецизност, одзив, $F1$ резултат и метрике подршке су описане у Табели 6.3.

Табела 6.3: Детаљне метрике по класама за најбољи модел *XG-BPSO* метахеуристике - Експеримент минимизације грешке класификације.

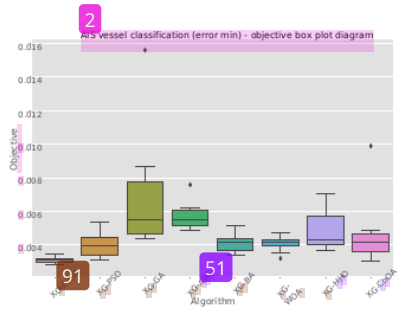
Class	*precision*	*recall*	*f1-score*	*support*
Cargo	0.998406	0.999527	0.998966	16921
Dredging	0.987097	0.995662	0.991361	461
Fishing	1.000000	0.994460	0.997222	1083
HSC	0.985207	0.994003	0.989599	335
Law enforcement	1.000000	1.000000	1.000000	136
Military	0.987085	0.988909	0.987996	541
Passenger	0.998011	0.995370	0.996689	1512
Pilot	1.000000	0.996139	0.998066	259
Pleasure	1.000000	1.000000	1.000000	7
Port tender	1.000000	1.000000	1.000000	5
Reserved	0.937500	0.849057	0.891089	53
SAR	1.000000	0.933333	0.965517	45
Sailing	0.900000	0.642857	0.750000	14
Tanker	0.998580	0.997730	0.998155	7050
Towing	0.880000	0.904110	0.891892	73
Towing long/wide	0.875000	0.980000	0.924528	50
Tug	0.995272	0.988263	0.991755	852
accuracy	0.997211	0.997211	0.997211	0.9972106
macro avg	0.973068	0.956438	0.963108	29397
weighted avg	0.997223	0.997212	0.997194	29397

Најбоље пронађене комбинације хиперпараметара за све метахеуристике су приказане у Табели 6.4.

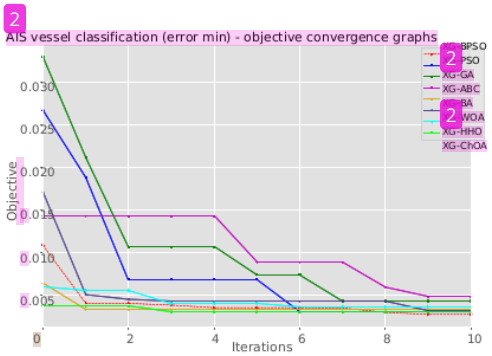
Табела 6.4: Најбољи селектовани хиперпараметри вишекласне класификације - Експерименти минимизације грешке класификације.

Method	*Learning Rate*	*Min child weight*	*Subsample*	*Colsample by tree*	*Max Depth*	*Gamma*
XG-BPSO	0.593383	1.000000	0.984577	1.000000	10	0.000000
XG-PSO	0.716738	2.148387	1.000000	0.931689	10	0.239401
XG-GA	0.741573	1.000000	0.944507	0.685084	9	0.382662
XG-ABC	0.709378	1.860859	0.684326	0.710326	10	0.000000
XG-BA	0.724748	3.134685	1.000000	1.000000	10	0.800000
XG-WOA	0.770394	3.170319	1.000000	1.000000	10	0.279516
XG-HHO	0.688403	2.552526	1.000000	1.000000	10	0.751155
XG-ChOA	0.642902	1.146574	1.000000	0.931855	10	0.637534

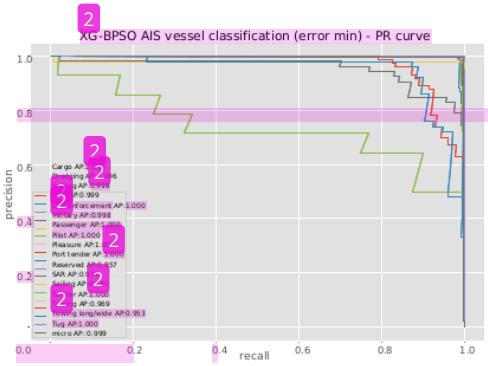
Визуелна анализа резултата експеримената класификације пловила за минимизацију грешке је приказана Сликом 6.3.



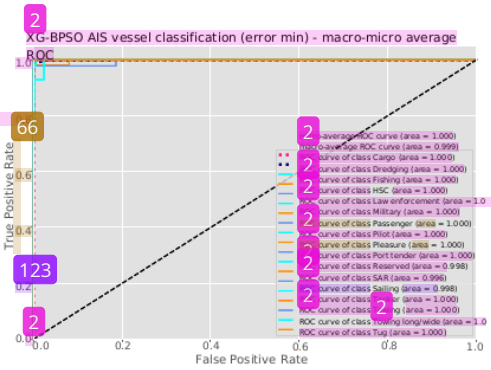
(а) Блок дијаграм циљне функције



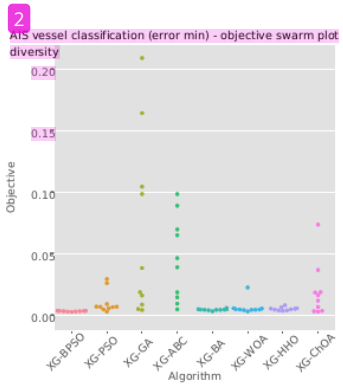
(б) Конвергенција циљне функције



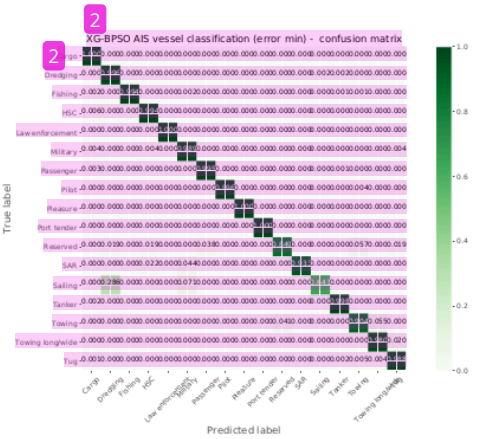
(в) Крива прецизности и одзива



(г) Крива карактеристике пријемника



(д) Разноликост података



(ђ) Конфузиона матрица

Слика 6.3: Дијаграми класификације пловила - минимизација грешке.

Резултати експеримената класификације пловила где је за циљну функцију упо-

требљен Коенов Капа параметар су приказани у Табели 6.5.

Табела 6.5: Укупни резултати циљне функције (Коенова Капа) - Експерименти максимизације Коеновог Капа параметра.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
XG-BPSO	**0.995561**	**0.993814**	**0.994878**	**0.994970**	**0.000531**	**2.82E-07**
XG-PSO	0.994379	0.990388	0.993022	0.993561	0.001328	1.76E-06
XG-GA	0.994098	0.982394	0.990003	0.990381	0.003349	1.12E-05
XG-ABC	0.992582	0.983179	0.989497	0.990521	0.002917	8.51E-06
XG-BA	0.994660	0.990496	0.993149	0.993396	0.001194	1.42E-06
XG-WOA	0.994660	0.989709	0.993029	0.993369	0.001442	2.08E-06
XG-HHO	0.994715	0.991679	0.993612	0.993901	0.000850	7.23E-07
XG-ChOA	0.994379	0.988417	0.992685	0.993227	0.001796	3.23E-06

Резултати индикаторске функције за експерименте класификације пловила за грешку класификације су приказани у Табели 6.6.

Табела 6.6: Укупни резултати индикаторске функције (грешка класификације) - Експерименти максимизације Коеновог Капа параметра.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
XG-BPSO	**0.002687**	**0.003742**	**0.003100**	**0.003045**	**0.000321**	**1.03E-07**
XG-PSO	0.003402	0.005817	0.004222	0.003895	0.000803	6.46E-07
XG-GA	0.003572	0.010647	0.006047	0.005817	0.002024	4.10E-06
XG-ABC	0.004490	0.010171	0.006353	0.005732	0.001763	3.11E-06
XG-BA	0.003232	0.005749	0.004146	0.003997	0.000721	5.21E-07
XG-WOA	0.003232	0.006225	0.004218	0.004014	0.000871	7.59E-07
XG-HHO	0.003198	0.005035	0.003865	0.003691	0.000514	2.64E-07
XG-ChOA	0.003402	0.007008	0.004426	0.004099	0.001086	1.18E-06

Детаљне метрике за експерименте класификације пловила са Коеновим Капа параметром као циљном функцијом су приказане у Табели 6.7.

Табела 6.7: Детальне метрике по класама за најбољи модел *XG-BPSO* метахеуристике - Експерименти максимизације Коеновог Капа параметра.

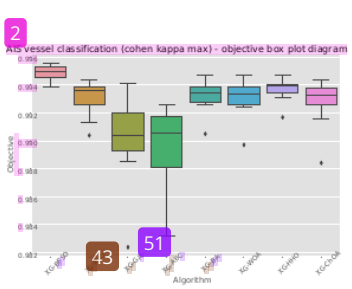
Class	*precision*	*recall*	*f1-score*	*support*
Cargo	0.998818	0.999114	0.998966	16921
Dredging	0.984979	0.995662	0.990291	461
Fishing	1.000000	0.995383	0.997686	1083
HSC	0.979412	0.994030	0.986667	335
Law enforcement	1.000000	0.985294	0.992593	136
Military	0.992647	0.998152	0.995392	541
Passenger	0.999336	0.996032	0.997681	1512
Pilot	0.992337	1.000000	0.996154	259
Pleasure	1.000000	1.000000	1.000000	7
Port tender	1.000000	1.000000	1.000000	5
Reserved	0.921569	0.886792	0.903846	53
SAR	1.000000	0.888889	0.941176	45
Sailing	1.000000	0.714286	0.833333	14
Tanker	0.997732	0.998298	0.998015	7050
Towing	0.942857	0.904110	0.923077	73
Towing long/wide	0.842105	0.960000	0.897196	50
Tug	0.994097	0.988263	0.991171	852
accuracy	0.997313	0.997313	0.997313	0.997313
macro avg	0.979170	0.959077	0.967250	29397
weighted avg	0.997346	0.997313	0.997303	29397

Најбољи селектовани хиперпараметри од сваке метахеуристке за експерименте максимизације Коеновог Капа параметра су описани у Табели 6.8.

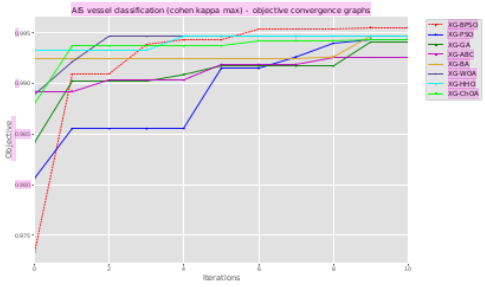
Табела 6.8: Најбољи селектовани хиперпараметри вишекласне класификације - Експерименти максимизације Коеновог Капа параметра.

ethod	*Learning Rate*	*Min child weight*	*Subsample*	*Colsample by tree*	*Max Depth*	*Gamma*
XG-BPSO	0.828630	2.737313	1.000000	0.868831	10	0.071057
XG-PSO	0.744281	3.435928	1.000000	0.872910	10	0.785282
XG-GA	0.730659	3.499590	1.000000	0.872590	10	0.256810
XG-ABC	0.681233	1.000000	1.000000	0.903165	9	0.550955
XG-BA	0.900000	6.501873	1.000000	0.871536	10	0.000000
XG-WOA	0.813991	3.414476	1.000000	0.860721	10	0.800000
XG-HHO	0.707112	2.440472	1.000000	1.000000	10	0.260687
XG-ChOA	0.734595	1.990239	1.000000	0.874517	10	0.684537

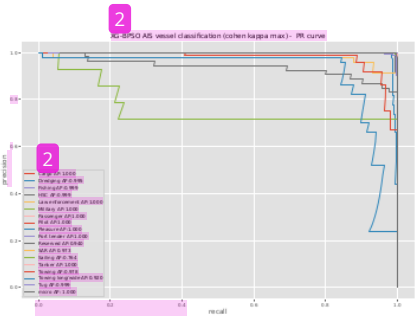
Визуелна анализа резултата експеримената класификације пловила за максимизацију Коеновог Капа параметра Сликом 6.4.



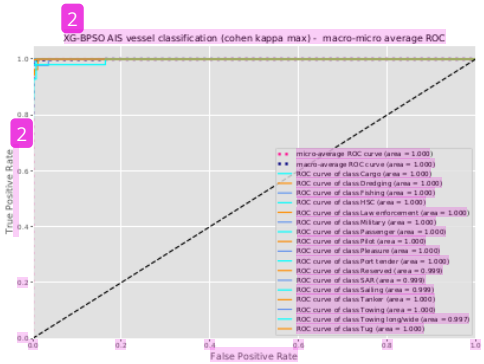
(а) Блок дијаграм циљне функције



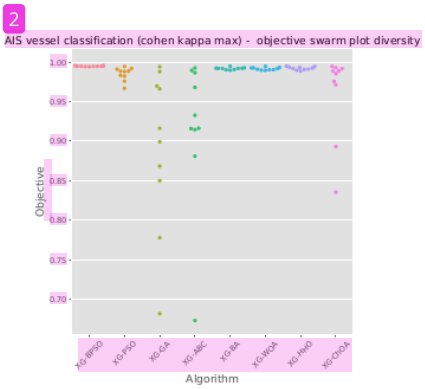
(б) Конвергенција циљне функције



(в) Крива прецизности и одзива



(г) Крива карактеристике пријемника



(д) Разноликост података



(ђ) Конфузиона матрица

Слика 6.4: Дијаграми класификације пловила - максимизација Коеновог Капа параметра.

6.2.3.3 Други експеримент: предвиђање путање пловила

Други експеримент се односи на предвиђање путање пловила и овај проблем се категоризује као мултиваријантно предвиђање са следећим обележјима ['LAT', 'LON', 'SOG', 'COG', 'Heading']. Циљне променљиве су ['LAT', 'LAN']. Валидациони сет није употребљен док су скупови за тренинг и тест у односу 50/50.

Најбоље резултате за метрику најбољег резултата је остварио *LSTM-BPSO*, док је други најбољи резултат имао оригинални *PSO*. Најмању вредност у категорији најгорег резултата је имао *LSTM-BA*, док је *LSTM-BPSO* постигао и најбоље резултате за средњу вредност и медијану. Најбоље резултате за стандардну девијацију и варијансу је постигао *LSTM-BA*. Описани резултати су приказани у Табели 6.9.

Табела 6.9: Укупни резултати циљне функције - Експерименти предвиђања путање пловила.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
LSTM-BPSO	**3.30E-05**	8.75E-05	**5.10E-05**	**3.23E-05**	1.90E-05	3.62E-10
LSTM-PSO	3.83E-05	1.04E-04	6.72E-05	6.11E-05	2.84E-05	8.06E-10
LSTM-GA	5.82E-05	1.25E-04	1.01E-04	1.14E-04	2.83E-05	8.03E-10
LSTM-ABC	5.13E-05	9.15E-05	6.85E-05	5.75E-05	1.79E-05	3.20E-10
LSTM-BA	4.86E-05	**6.04E-05**	5.28E-05	5.22E-05	**4.80E-06**	**2.30E-11**
LSTM-WOA	6.45E-05	1.02E-04	8.36E-05	8.65E-05	1.48E-05	2.19E-10
LSTM-HHO	5.17E-05	8.29E-05	6.73E-05	6.04E-05	1.21E-05	1.47E-10
LSTM-ChOA	5.22E-05	1.01E-04	7.22E-05	6.83E-05	1.82E-05	3.32E-10

Најбоље резултате за R^2, *MSE* и *RMSE* је постигао *LSTM-BPSO*, док је *LSTM-PSO* остварио најбоље резултате за метрике *IoA*, *MAE*, и *EDE* како је и приказано у Табели 6.10.

Табела 6.10: Детаљне метрике за најбољи модел - Експерименти предвиђања путање пловила.

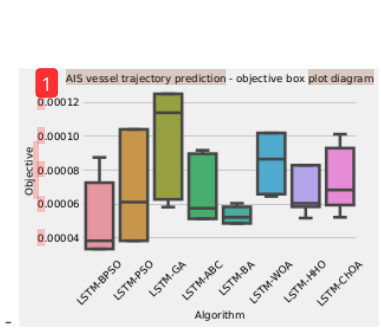
Method	R^2	*MAE*	*MSE*	*RMSE*	*IoA*	*EDE*
LSTM-BPSO	**0.997921**	0.007313	**0.000098**	**0.009985**	0.999769	0.012385
LSTM-PSO	0.997912	**0.006937**	0.000099	0.009987	**0.999774**	**0.011048**
LSTM-GA	0.997311	0.008302	0.000128	0.011299	0.999708	0.013484
LSTM-ABC	0.997361	0.008045	0.000126	0.011216	0.999711	0.012139
LSTM-BA	0.997554	0.007366	0.000117	0.010794	0.999735	0.011591
LSTM-WOA	0.997002	0.009676	0.000142	0.011930	0.999672	0.015390
LSTM-HHO	0.997090	0.009045	0.000139	0.011795	0.999679	0.013913
LSTM-ChOA	0.997654	0.008605	0.000111	0.010547	0.999744	0.013513

Најбољи скупови параметара које је свака од тестираних метахеуристика остварила су приказани у Табели 6.11.

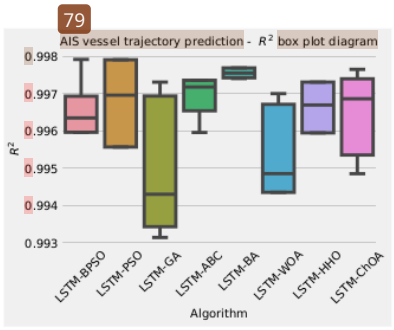
Табела 6.11: Најбољи селектовани хиперпараметри - Експерименти предвиђања путање пловила.

Method	*Learning Rate*	*Dropout*	*Epochs*	*Layers*	*Neurons layer 1*	*Neurons layer 2*
LSTM-BPSO	0.000656	0.005000	50	1	27	/
LSTM-PSO	0.000125	0.020000	40	1	31	/
LSTM-GA	0.006729	0.060654	39	1	22	/
LSTM-ABC	0.004661	0.105701	44	2	16	24
LSTM-BA	0.000086	0.005000	40	1	32	/
LSTM-WOA	0.003918	0.154873	34	1	27	/
LSTM-HHO	0.001845	0.093174	41	1	26	/
LSTM-ChOA	0.002802	0.144773	46	1	20	/

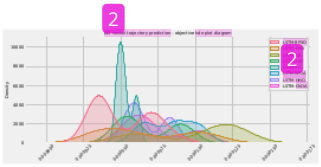
Визуелна анализа резултата експеримената предвиђања путање пловила је приказана Сликом 6.5 на којој су приказани и дијаграми процене густине језгра (енгл. - *kernel density estimation - KDE*).



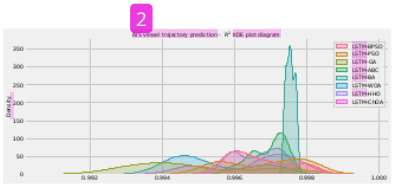
(а) Блок дијаграм циљне функције



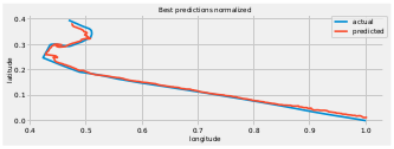
(б) R^2 блок дијаграм



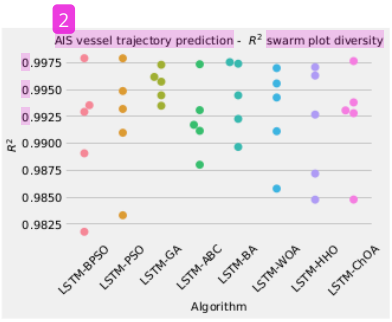
(в) *KDE* дијаграм циљне функције



(г) R^2 *KDE* дијаграм



(д) Нормализовани најбољи резултати



(ђ) R^2 разноликост података

Слика 6.5: Дијаграми предвиђања путање пловила.

6.2.4 Статистичка евалуација

Ради даље анализе резултата симулације као и утврђивања њихове статистичке важности, најбољи резултати из 30 независних покретања су прикупљени за сваку метахеуристику. Подаци су процењивани као серије тачака података. Прво је евалуација обављена параметарским тестирањем посматрајући независност, нормалности и хомоскедастичности варијансе у подацима [65]. Услов независности је постигнут генерисањем насумичних скупова решења при сваком независном покретању.

Услов нормалности је проверен путем Шапиро-Вилк (енгл. - *Shapiro-Wilk*) теста са индивидуалном анализом проблема [66], за сва три експеримента који укључују класификацију пловила на два начина (минимизација грешке и максимизације Коенове Капе) и предвиђање путање пловила. Шапиро-Вилк тест је примењен независно за сва три случаја и као резултат се добијају p вредности за сваки тест. Ове вред-

ности су у сва три случаја биле испод 0.05, указујући на могућност одбијања нулте хипотезе X0. Коначни закључак тестова је да сва три експеримента не прате нормалну расподелу. За случајеве који су били преко границе је примењен подебљани фонт.

Резултати Шапиро-Вилк теста за сва три случаја су приказани у Табели 6.12.

Табела 6.12: Шапиро-Вилк тестови нормалности.

Problem	*BPSO*	*PSO*	*GA*	*ABC*	*BA*	*WOA*	*HHO*	*ChOA*
Vessel Error	0.032	0.041	0.046	0.018	0.021	0.033	0.035	0.040
Vessel Kappa	0.043	0.015	0.044	0.031	0.032	0.038	0.021	0.035
Trajectory	0.011	0.003	0.015	0.009	0.019	0.006	0.010	0.013

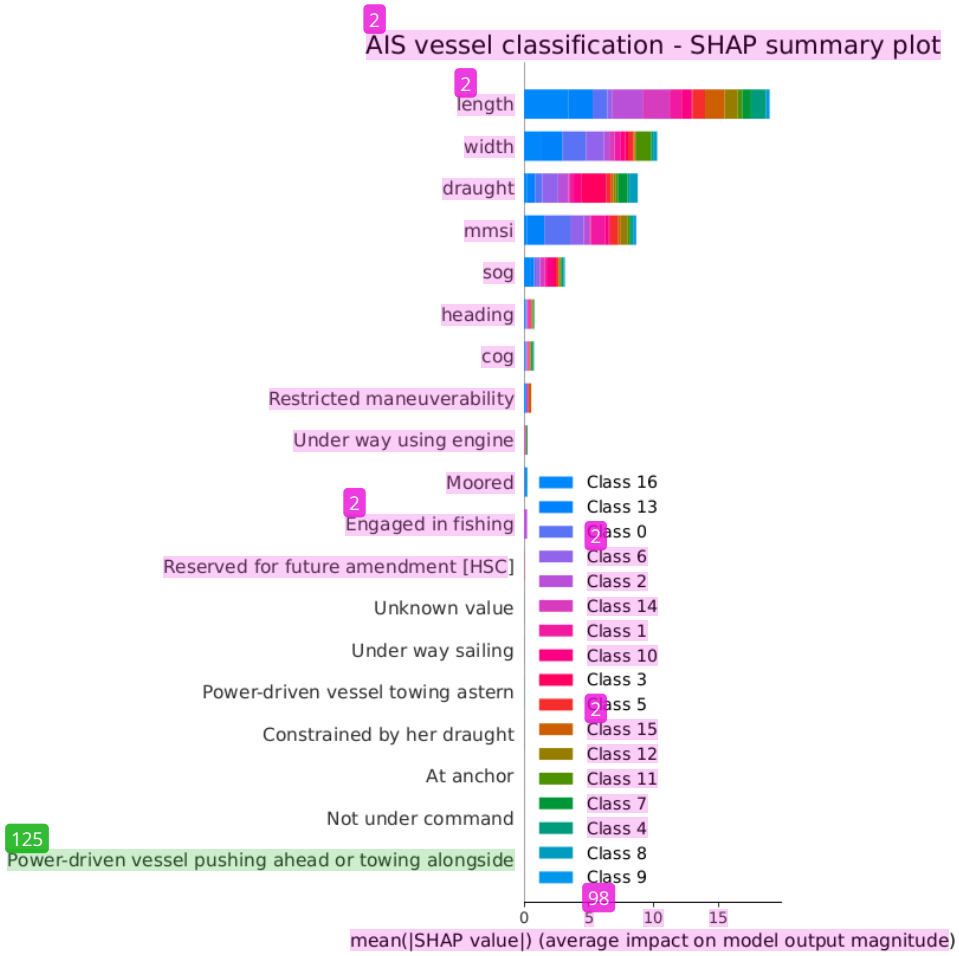
Због исхода тестова нормалности параметарски тестови нису примењени. Следећи корак је била примена Вилкоксоновог (енгл. - *Wilcoxon*) теста упарених узорака [67], који је примењен над истим серијама података које се састоје од најбољих вредности постигнутих од стране сваке метахеуристике. Контролни алгоритам у овој врсти теста је *BPSO*. За сва три случаја p вредности нису биле веће од 0.05. Узимајући $\alpha = 0.1$ за праг важности, резултати показују да је *BPSO* статистички значајно супериоран. За случај када се за праг важности узме $\alpha = 0.05$, *BPSO* је знатно бољи од свих решења осим од *BA* у експерименту предвиђања путање и овај резултат је означен подебљаним фонтом. Укупни резултати Вилкоксоновог теста су у Табели 6.13.

Табела 6.13: Резултати Вилкоксоновог теста упарених узорака који показују p-вредности за сва три експеримента (*BPSO* у односу на остале).

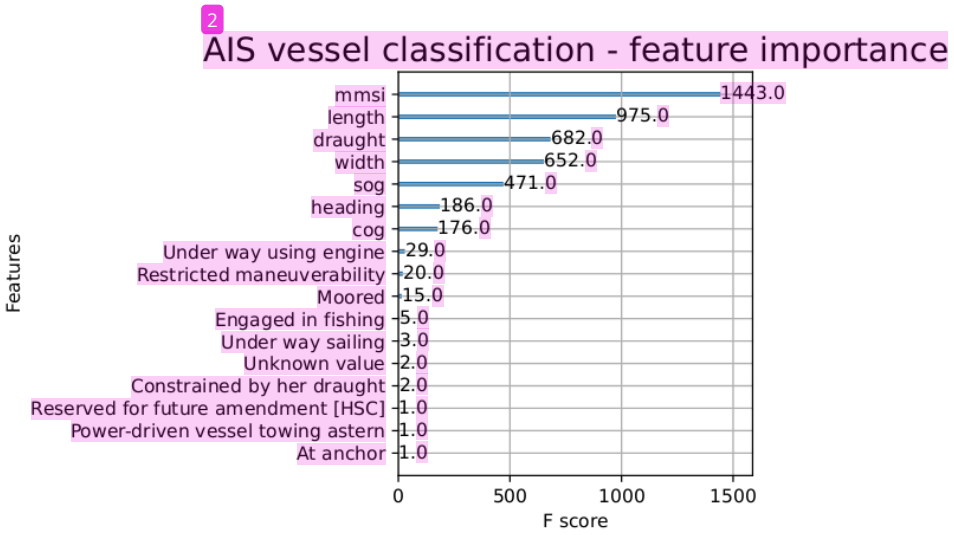
Problem / p-values	*PSO*	*GA*	*ABC*	*BA*	*WOA*	*HHO*	*ChOA*
Vessel Error	0.042	0.003	0.018	0.037	0.04	0.027	0.031
Vessel Kappa	0.043	0.027	0.023	0.045	0.044	0.047	0.036
Trajectory	0.045	0.007	0.041	**0.054**	0.024	0.044	0.032

6.2.5 Интерпретација резултата најбољих модела

Интерпретација модела је фаза од пресудног значаја јер ако није јасно због чега је неки модел бољи онда је и шанса за примену и модела и технике која га потенцијално чини бољим проблематична. Модерна литература препознаје три главна начина за интерпретацију модела, а избор који је популаран међу истраживачима су Шепли адитивна објашњења (енгл. - *SHapley Additive exPlanations - SHAP*) [68]. Ова метода примењује принципе из теорије игара како би важност обележја била одређена. Додатно, алгоритам *XGBoost* има уграђену методу за одређивање важности обележја која је базирана на Џини индексу нечистоће (енгл. - *gini impurity index*) [69]. Разумевање и поређење важности помажу у унапређивању предикције и резултати су дати Сликом 6.6 и Сликом 6.7.

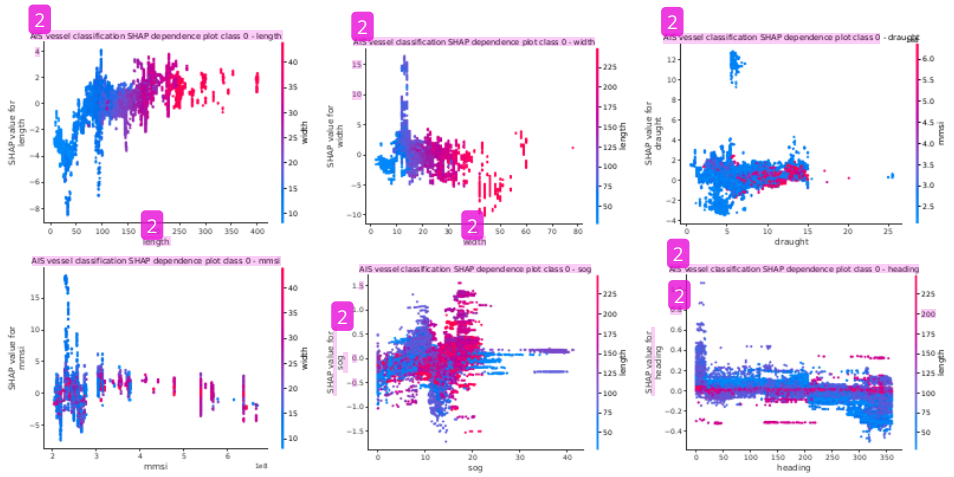


Слика 6.6: Сумарна важност обележја одређена путем *XGBoost* и *SHAP*.

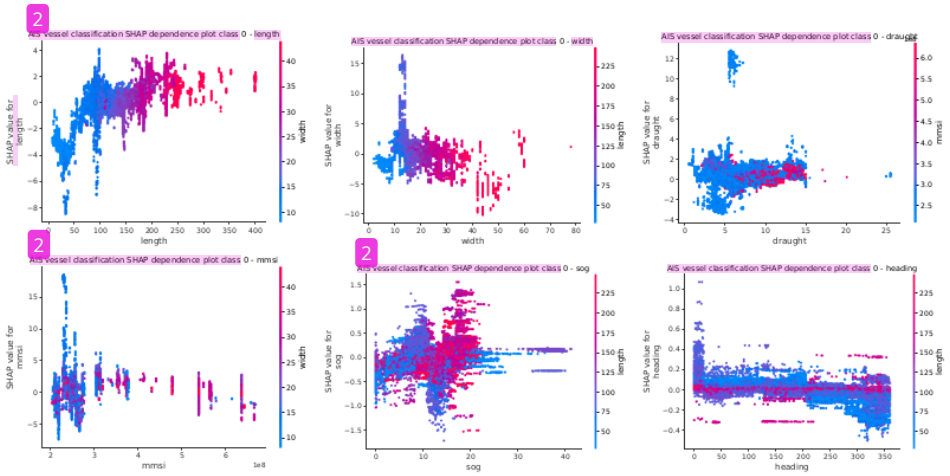


Слика 6.7: Важност обележја одређена путем *XGBoost* и *SHAP*.

Анализа важности обележја показује да дужина пловила има важну улогу у тачној класификацији за све типове пловила. Друго најважније обележје је ширина пловила, па затим газ и на крају *MMSI*. Параметри који су променљивији од пловила до пловила као што су брзина изнад дна, правац и курс изнад дна имају мање важну улогу. Са друге стране, брзина изнад дна има важнију улогу што је вероватно због ограничења брзине код масивнијих пловила. Важно је напоменути у анализи обележја да она имају различите утицаје на различите одлуке. Шест најбољих обележја за одређивање теретних (енгл. - *cargo*) пловила су описани Сликом 6.8. Шест најважнијих обележја за одлучивање о тегљачима су илустровани Сликом 6.9.



Слика 6.8: Графикони важности обележја за теретна пловила.



Слика 6.9: Графикони важности обележја за тегљаче.

Анализирајући претходне фигуре, може се утврдити да је дужина пловила важнија за теретна пловила него за тегљаче, као и ширина пловила. Са друге стране, газ је битнији код тегљача.

6.3 Експерименти препознавања грешке у програмском коду

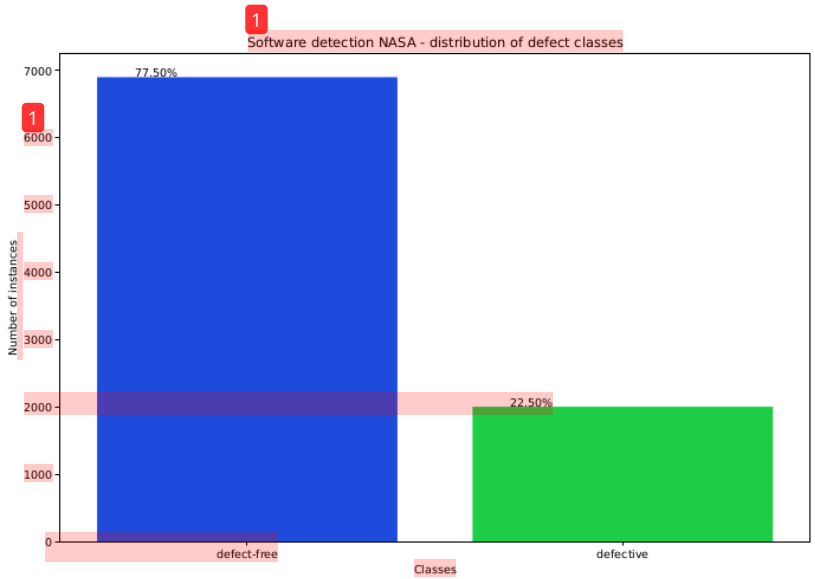
Унапређења *PSO* алгоритма у *MPPSO* алгоритам су такође потврђена на својој групи експеримената. Примењена су два различита скупа података у различитим комбинацијама са моделима вештачке интелигенције. Оптимизовани су *CNN*, *AdaBoost* и *XGBoost*. Експерименти се могу поделити и по дубини слојева модела који је примењен. Извршени су и експерименти са *NLP* методама. Експерименти другог слоја предложеног радног оквира су такође извршени и са применом *NLP* метода. Енкодовање је извршено помоћу *TF-IDF* векторизације са ограничењем на 1000 обележја. Затим су векторизоване вредности употребљене као улаз другог слоја радног оквира у којем се користе *AdaBoost* или *XGBoost* као класификатори. Оптимизација је извршена над овим класификационим моделима са циљем унапређивања општих перформанси избором најбољих параметара.

6.3.1 Поставка експеримената

Другу групу експеримената чине две врсте симулације. Експерименти се даље могу још једном поделити, где прву групу чине скупови података *KC1, JM1, CM1, KC2* и *PC1*. Наведени скупови података су део групе скупова података из Националне агенције за аеронаутику и свемир (НАСА) репозиторијума за детекцију дефекта у софтверу. Инстанце у скуповима података који чине овај скуп су различити модели софтвера, а обележја описују квалитет кода. Томас Ц. Мекејб [70] (енгл. - *Thomas J. McCabe*) и Морис Хауард Халстед (енгл. - *Maurice Howard Halstead*) [71] метрике су примењене на 22 обележја. Мекејб приступ вреднује мање компликован код кроз

циљ смањивања путања, док Халстед користи технику бројања елемената кода јер како принцип овог приступа налаже - више кода значи и већу шансу за грешком. Симулације су извршене по двослојном принципу.

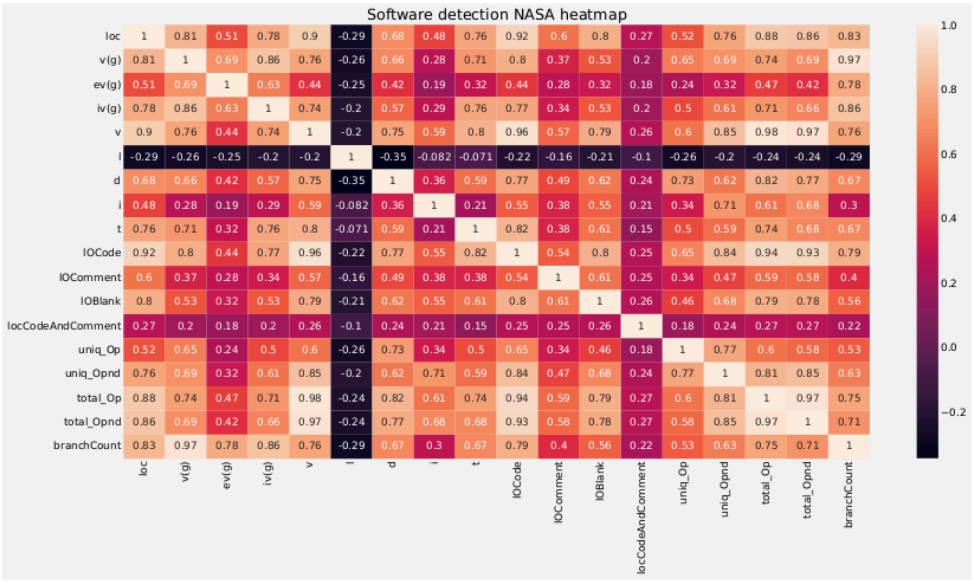
Баланс класа у примењеним скуповима података је приказан Сликом 6.10.



Слика 6.10: Дистрибуција класа у НАСА скупу података.

Први скуп података нема добар баланс класа јер само 22.5% случајева има грешку у софтверу док осталих 77.5% представља код који нема грешке.

Корелациона матрица обележја је приказана Сликом 6.11.



Слика 6.11: Корелација обележја у HACA скупу података.

Други скуп података се састоји од око 1000 примера *Python* програма, које је са различитих форума сакупила програмерска заједница. Скуп података је јавно доступан https://paperswithcode.com/dataset/mbpp и приступљено му је 01.08.2024. Проблеми су на почетничком нивоу и састоје се од описа задатака, решења и 3 аутоматизована тест случаја. Генеративни модел *GPT2* је употребљен за генерисање примера проблема на основу описа. Генерисани случајеви који су прошли и пали бар по један тест случај су прикупљани и комбиновани у спојени скуп података. Финални скуп се састоји од 10000 примера где су класе избалансиране.

Оба скупа података су раздвојена у тренинг и тест део и то у односу 70/30. Стандардне метрике за евалуацију класификације су примењене као што су тачност, одзив, $F1$ резултат и прецизност. Метјузов коефицијент корелације (енгл. - *Matthews correlation coefficient* - *MCC*) је употребљен као циљна функција и описан Једначином 6.12.

$$\text{MCC} = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}, \tag{6.12}$$

где су истините позитивне вредност приказане као TP истините негативне вредности као NP, лажно позитивне вредности као FP и лажно негативне вредности као FN. Додатно је примењена и индикаторска функција за шта је примењена стопа грешке.

Циљ метахеуристика је избор оптималних параметара за максимизацију *MCC* резултата. Први слој чини оптимизација *CNN* параметара чије границе су приказне Табелом 6.14.

Други слој оптимизације је предодређен да обрађује директне излазе из *CNN*. Последњи је густи слој (енгл. - *dense layer*) који бележи све класификације узорака из скупа података. Узорци су подељени у односу 70/30, где је већи удео одвојен за тренирање.

Табела 6.14: Опсези *CNN* параметара за оптимизацију.

Parameter	*Range*
Learning Rate	0.0001 - 0.0030
Dropout	0.05 - 0.5
Epochs	10 - 20
Number of CNN Layers	1 - 3
Number of Dense Layers	1 - 3
Number of Neurons in layer	32 - 128

Следећи корак је примена ових података за оптимизацију *AdaBoost* и *XGBoost* модела. Границе параметара за ове моделе су приказане Табелом 6.15.

Табела 6.15: Опсези *AdaBoost* параметара за оптимизацију.

Parameter	*Range*
Count of estimators	$[10, 30]$
Depth	$[1, 5]$
Learning rate	$[0.01, 2]$

Опсези вредности за оптимизоване хиперпараметре *XGBoost* модела су приказани Табелом 6.16.

Табела 6.16: Опсези *XGBoost* параметара за оптимизацију.

Parameter	*Range*
Learning Rate	0.1 - 0.9
Minimum child weight	1 - 10
Subsample	0.01 - 1.00
Col sample by tree	0.01 - 1.00
Max depth	3 - 10
Gamma	0.00 - 0.80

Неколико оптимизатора је увршteно у компаративну анализу и то су основна верзија *PSO* [3], *GA* [33], *VNS* [72], *ABC* [35], *BA* [36], *COLSHADE* [73] и *SCHO* [40].

Сваки од наведених оптимизатора је примењен са основном поставком параметара из радова у којима су представљени. Величина популације је подешена на 10 јединки, а укупан број итерација на 8. Симулације се извршавају у 30 независних покретања ради веродостојности експеримената.

За *NLP* симулације је примењен само други слој радног оквира. Текст који је употребљен као унос је енкодован путем *TF-IDF* технике и то до максимално 1000 токена. Ови улази су примењени за тренирање и евалуацију модела. Оптимизација је извршена путем другог слоја предложеног радног оквира, а упоредна анализа користи идентичне услове поређења као претходна симулација ради приказа флексибилности модела предложеног решења.

6.3.2 Симулације *CNN* класификације на првом слоју

Представљена *MPPSO* модификована метахеуристика показује најбоље резултате и остварила је 0.273280 за најбољу метрику и 0.262061 за најгору метрику. Такође је предложени алгоритам остварио најбоље резултате за средњу вредност од 0.268824 и 0.269873 за медијану. Најбољу стабилност је показао *BA* алгоритам, али је заостајао по другим критеријумима што представља слабост у домену експлорације и експлоатације. Ови резултати су приказани у Табели 6.17.

Табела 6.17: Исходи цильне функције *L1 CNN* модела.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
CNN-MPPSO	**0.273280**	**0.262061**	**0.268824**	**0.269873**	0.004194	1.76E-05
CNN-PSO	0.267168	0.253372	0.261106	0.260917	0.004766	2.27E-05
CNN-GA	0.265933	0.249726	0.258941	0.258989	0.005977	3.57E-05
CNN-VNS	0.265920	0.258217	0.262605	0.263118	0.002553	6.52E-06
CNN-ABC	0.269294	0.253197	0.261761	0.262521	0.005419	2.94E-05
CNN-BA	0.265823	0.258906	0.261940	0.261646	**0.002218**	**4.92E-06**
CNN-SCHO	0.268643	0.257094	0.261124	0.259722	0.003986	1.59E-05
CNN-COLSHADE	0.262284	0.255697	0.259373	0.260087	0.002277	5.18E-06

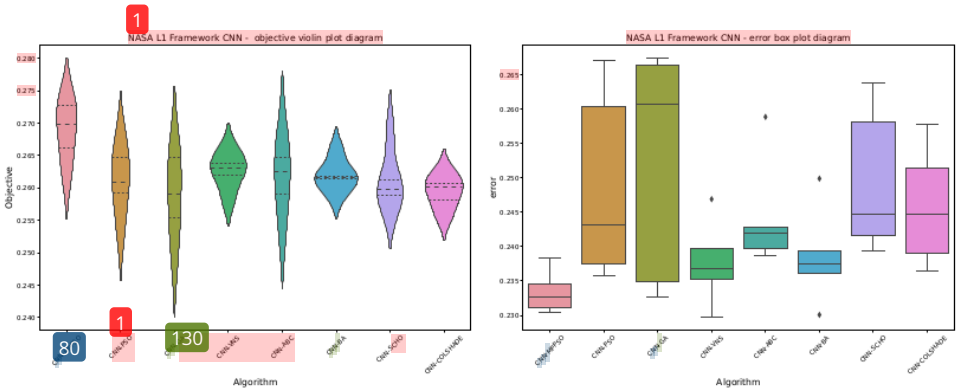
Резултати *CNN* на првом слоју за индикаторску функцију су дати у Табели 6.18.

Табела 6.18: Исходи индикаторске функције *L1 CNN* модела.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
CNN-MPPSO	**0.231201**	0.238309	**0.233446**	**0.232697**	**0.002810**	**7.89E-06**
CNN-PSO	0.243172	0.267116	0.248784	0.243172	0.012645	1.60E-04
CNN-GA	0.319865	0.260756	0.252450	0.260756	0.015398	2.37E-04
CNN-VNS	0.236813	0.239805	0.237710	0.236813	0.005652	3.19E-05
CNN-ABC	0.242050	0.242798	0.244444	0.242050	0.007371	5.43E-05
CNN-BA	0.239431	**0.230079**	0.238608	0.237561	0.006460	4.17E-05
CNN-SCHO	0.244669	0.263749	0.249532	0.244669	0.009629	9.27E-05
CNN-COLSHADE	0.239057	0.244669	0.245866	0.244669	0.007860	6.18E-05

Предложени алгоритам, *MPSO* показује сличне резултате као на претходним тестовима и поново има најбољи резултат од 0.231201. Средња вредност и медијана такође имају најбоље вредности у случају предложеног алгоритма и то 0.233446 за средњу вредност, а 0.232697 за медијану. Висока стабилност је примећена и код основне верзије алгоритма, али предложена модификована верзија даје боље резултате.

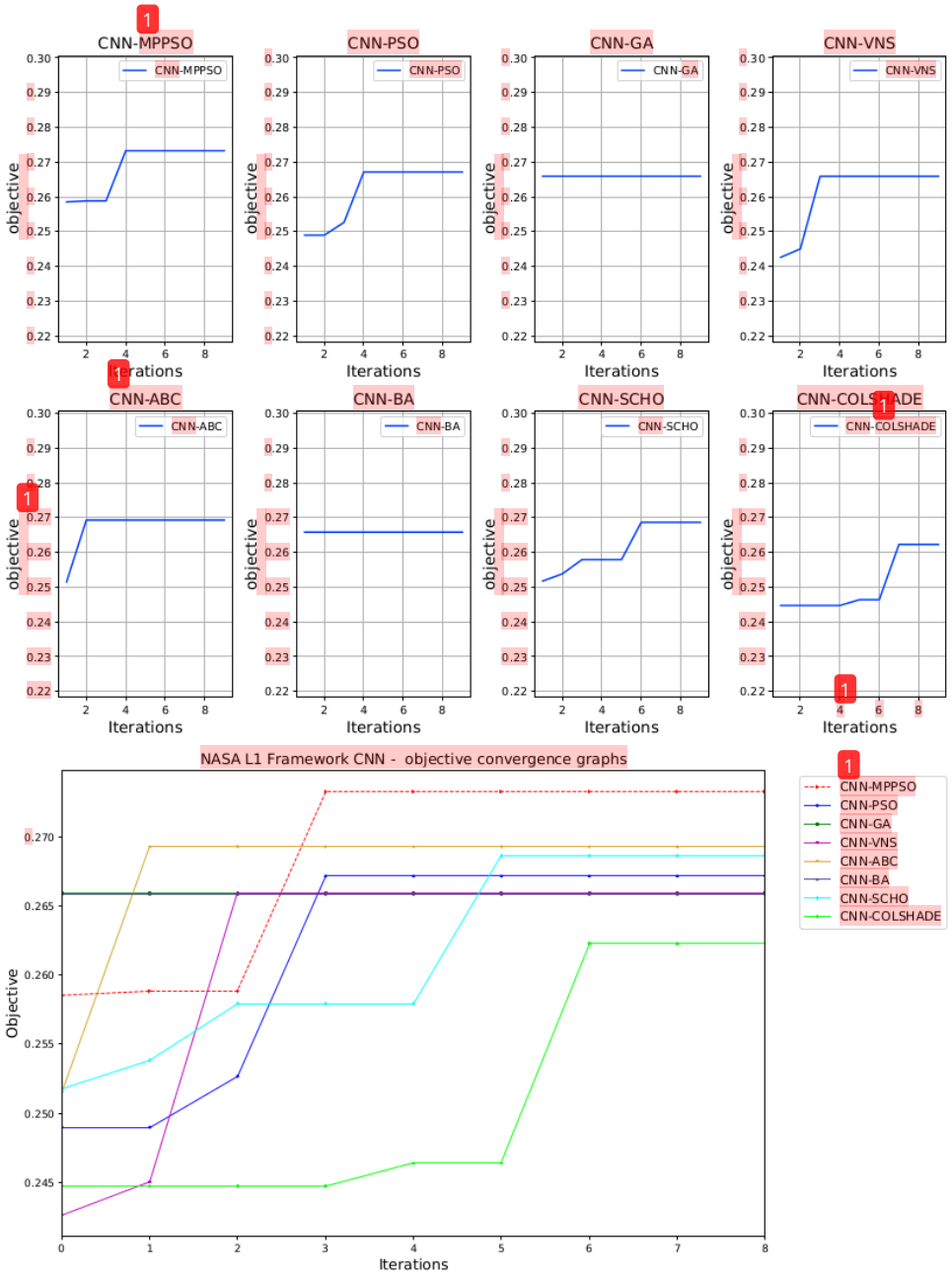
Визуелизација стабилности тестираних оптимизатора је осликана Сликом 6.12.



Слика 6.12: Дистрибуције цильне и индикаторске функције *L1 CNN* модела.

Алгоритми као што су *VNS*, *BA* и *COLSHADE* показују већу стабилност од предложеног решења, али заостају у квалитету пронађених решења. Може се претпоставити да се ово понашање изражава због тога што се ови алгоритми фокусирају на локални оптимум више него на проналажење глобално најбољег решења. Узимајући у обзир метахеуристичка решења и принцип по којем раде, ово значи да недостаци у фази експлорације доводе до мање погодних решења. Приказани резултати потврђују да је унапређење у односу на основно решење постигнуто јер предложени алгоритам не само да има боље резултате него његова основна верзија има и бољу стабилност у предикцијама.

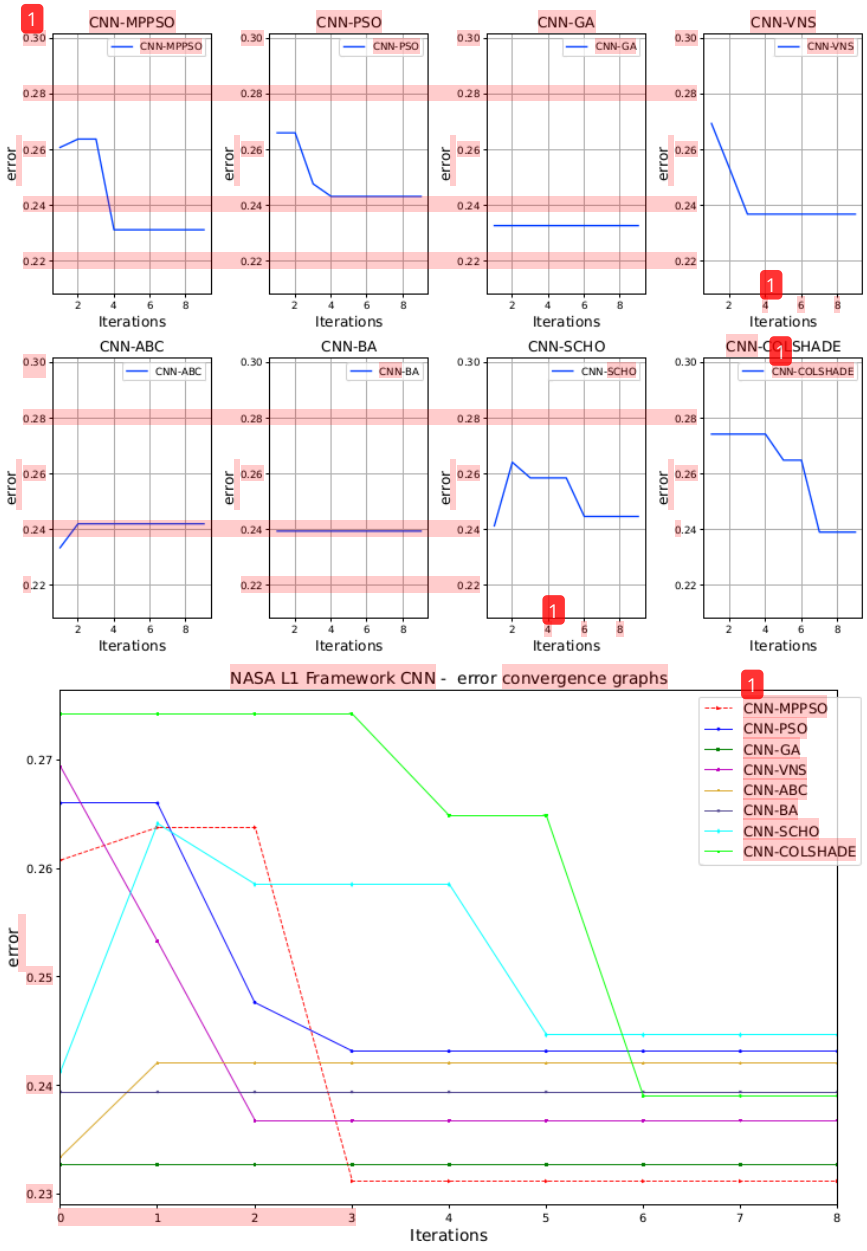
Стопе конвергенције показују важне информације о понашању оптимизатора. Стопе конвергенције за цильну функцију су приказане Сликом 6.13.



Слика 6.13: Конвергенција цильне функције *L1 CNN* модела.

Предложени оптимизатор достиже завидне перформансе проналажења најбољег решења већ до 8. итерације. Остали оптимизатори стагнирају пре него што остваре добар резултат. Из овог понашања се може закључити да су измене основне верзије алгоритма донеле унапређења по питању експлорације.

Стопе конвергенције за индикаторску функцију су приказане Сликом 6.14.



Слика 6.14: Конвергенција индикаторске функције *L1 CNN* модела.

Слична опажања су присутна и са конвергенцијом индикаторске функције као и код конвергенције циљне функције. Узимајући у обзир да индикаторска функција није била предмет оптимизације, приметан је благи пад перформанси у првим ите-

рацијама оптимизације. Најбољи резултат је зато пронађен од стране предложене методе до треће итерације.

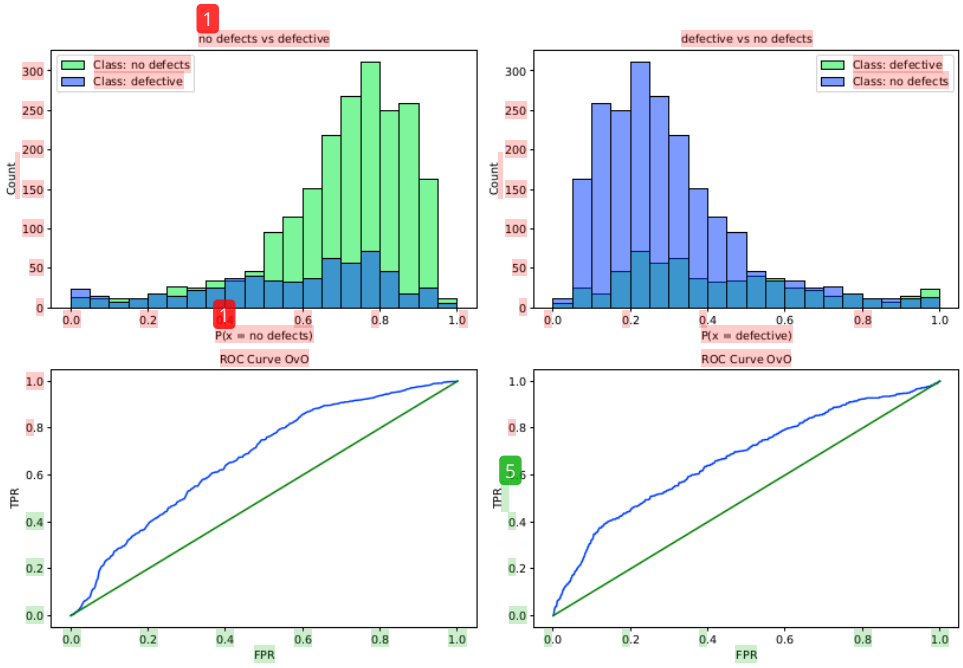
Детаљна поређења најбоље оптимизованих модела од сваког алгоритма су представљена Табелом 6.19.

Табела 6.19: Детаљно поређење модела са најбољим учинком *L1 CNN* симулација.

method	*metric*	*no error*	*error*	*accuracy*	*macro avg*	*weighted avg*
CNN-MPPSO	*precision*	0.826009	**0.480813**	**0.768799**	**0.653411**	**0.748395**
	recall	**0.888996**	0.354409	**0.768799**	0.621703	**0.768799**
	f1-score	**0.856346**	0.408046	**0.768799**	0.632196	**0.755550**
CNN-PSO	*precision*	0.829777	0.452611	0.756828	0.641194	0.744975
	recall	0.863417	0.389351	0.756828	0.626384	0.756828
	f1-score	0.846263	0.418605	0.756828	0.632434	0.750108
CNN-GA	*precision*	0.768653	0.188725	0.680135	0.478689	0.638262
	recall	0.840251	0.128120	0.680135	0.484185	0.680135
	f1-score	0.802859	0.152626	0.680135	0.477743	0.656660
CNN-VNS	*precision*	0.826304	0.465812	0.763187	0.646058	0.745250
	recall	0.879344	0.362729	0.763187	0.621036	0.763187
	f1-score	0.851999	0.407858	0.763187	0.629928	0.752138
CNN-ABC	*precision*	0.830014	0.455253	0.757950	0.642633	0.745752
	recall	0.864865	0.389351	0.757950	**0.627108**	0.757950
	f1-score	0.847081	0.419731	0.757950	0.633406	0.750995
CNN-BA	*precision*	0.827539	0.459959	0.760569	0.643749	0.744892
	recall	0.873069	0.372712	0.760569	0.622891	0.760569
	f1-score	0.849695	0.411765	0.760569	0.630730	0.751230
CNN-SCHO	*precision*	**0.830999**	0.450094	0.755331	0.640547	0.745356
	recall	0.859073	**0.397671**	0.755331	0.628372	0.755331
	f1-score	0.844803	**0.422261**	0.755331	**0.633532**	0.749798
CNN-COLSHADE	*precision*	0.826127	0.460084	0.760943	0.643105	0.743825
	recall	0.875965	0.364393	0.760943	0.620179	0.760943
	f1-score	0.850316	0.406685	0.760943	0.628501	0.750570
	support	2072	601			

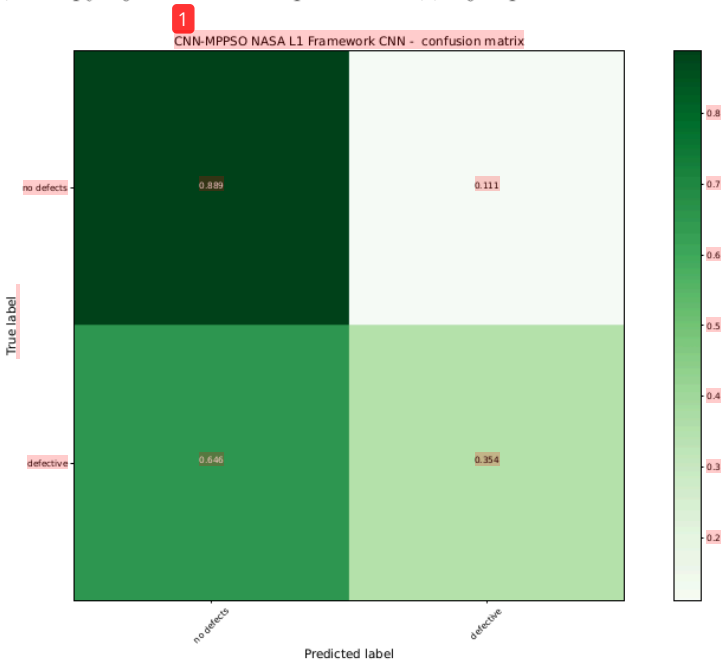
Представљено решење доминира категорије тачности и прецизности. Са друге стране, остали оптимизатори имају своје предности које се примећују у осталим категоријама што даље подржава *NFL* теорему.

ROC крива најбољег модела *MPPSO* алгоритма је приказана Сликом 6.15.



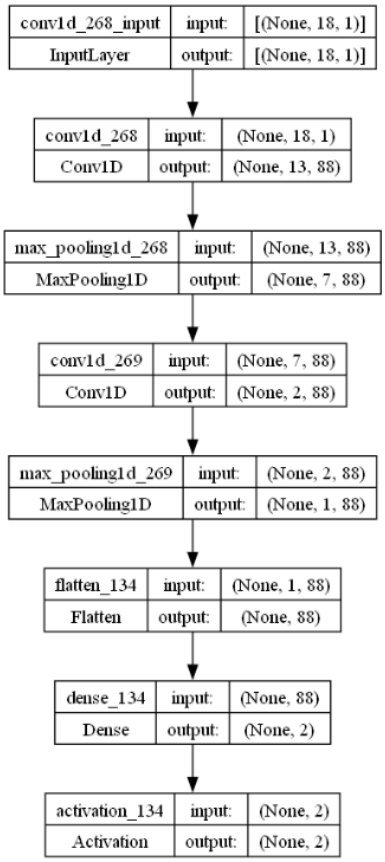
Слика 6.15: *ROC* крива *L1 CNN-MPPSO* оптимизованог модела.

Матрица конфузије за исти алгоритам и модел је приказана Сликом 6.16.



Слика 6.16: Матрица конфузије *L1 CNN-MPPSO* оптимизованог модела.

Нацрт најбољег модела је представљен Сликом 6.17.



Слика 6.17: Визуелизација *CNN-MPPSO* оптимизованог *L1* модела.

Оптимални параметри које су оптимизациони алгоритми одредили су представљени Табелом 6.20.

Табела 6.20: Селектовани параметри оптимизованих *CNN* модела.

Method	*Learning Rate*	*Dropout*	*Epochs*	*Layers CNN*	*Layers Dense*	*Neurons CNN*	*Neurons Dense*
CNN-MPPSO	2.44E-03	0.500000	84	1.0	1.0	88	88
CNN-PSO	2.80E-03	0.165591	56	1.0	1.0	97	83
CNN-GA	1.66E-03	0.475371	65	1.0	1.0	94	86
CNN-VNS	2.38E-03	0.500000	42	1.0	1.0	128	93
CNN-ABC	2.61E-03	0.370234	93	2.0	1.0	124	96
CNN-BA	3.00E-03	0.500000	70	2.0	2.0	108	103
CNN-SCHO	1.25E-03	0.197273	100	1.0	2.0	46	105
CNN-COLSHADE	1.88E-03	0.230465	86	2.0	2.0	73	99

6.3.3 Симулације *AdaBoost* класификације на другом слоју

Исходи *AdaBoost* оптимизације су приложени у Табели 6.21.

Табела 6.21: Исходи цильне функције *L2 AdaBoost* модела.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
CNN-AB-MPPSO	**0.282906**	**0.276185**	**0.279771**	**0.280600**	0.003092	9.56E-06
CNN-AB-PSO	**0.282906**	0.270598	0.276496	0.276185	0.003085	9.51E-06
CNN-AB-GA	**0.282906**	0.268442	0.277295	0.276351	0.002980	8.88E-06
CNN-AB-VNS	**0.282906**	0.269109	0.276079	0.276185	0.003662	1.34E-05
CNN-AB-ABC	**0.282906**	0.266412	0.273352	0.274076	0.004034	1.63E-05
CNN-AB-BA	0.276185	0.202141	0.262977	0.271189	0.019258	3.71E-04
CNN-AB-SCHO	**0.282906**	0.271101	0.276876	0.276185	**0.002530**	**6.40E-06**
CNN-AB-COLSHADE	**0.282906**	0.270156	0.275568	0.276185	0.003515	1.24E-05

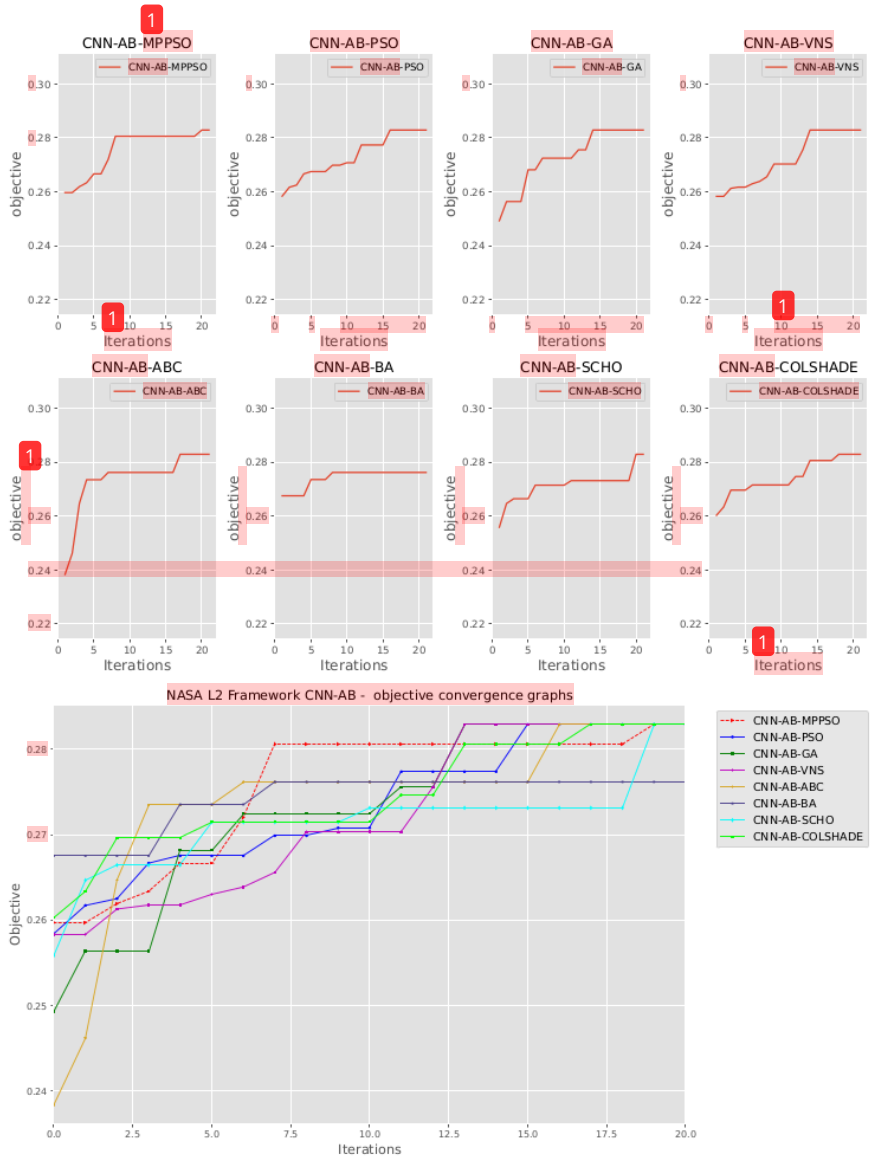
Предложени алгоритам постиже најбоље резултате за медијану, најгору и средњу вредност и то 0.280600, 0.276185, и 0.279771, респективно. По питању најбољих резултата, *MPPSO* и основни *PSO* постижу исте перформансе као и конкуренција. Највише нивое стабилности је постигао *SCHO*, али ово метахеуристичко решење се није показало у осталим категоријама.

Резултати симулација другог слоја *AdaBoost* оптимизације за индикаторску функцију су приказани у Табели 6.22.

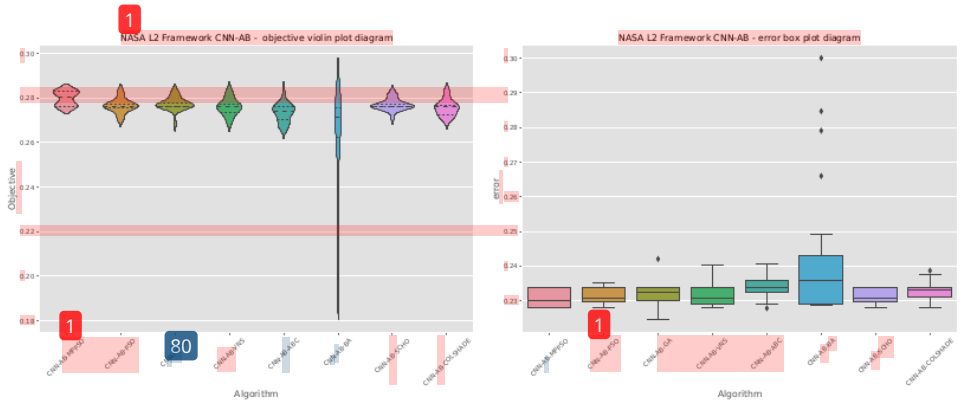
Табела 6.22: Исходи индикаторске функције *L2 AdaBoost* модела.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
CNN-AB-MPPSO	**0.227834**	0.233820	**0.230453**	**0.230079**	0.002661	7.08E-06
CNN-AB-PSO	**0.227834**	**0.228208**	0.231481	0.230827	0.002380	5.67E-06
CNN-AB-GA	**0.227834**	0.242050	0.231874	0.232323	0.003529	1.25E-05
CNN-AB-VNS	**0.227834**	0.239057	0.232080	0.230827	0.003587	1.29E-05
CNN-AB-ABC	**0.227834**	0.235316	0.234287	0.233820	0.003431	1.18E-05
CNN-AB-BA	0.233820	0.249158	0.244239	0.235690	0.020533	4.22E-04
CNN-AB-SCHO	**0.227834**	0.233446	0.231257	0.230640	**0.002208**	**4.87E-06**
CNN-AB-COLSHADE	**0.227834**	0.236813	0.232641	0.233071	0.002865	8.21E-06

Приказани резултати индикаторске функције указују на доминантност *BA* по питању стабилности, док предложено решење постиже најбоље резултате. Овај случај је специфичан јер су скоро сви алгоритми изједначини осим *BA* у категорији најбољег резултата. Стопе конвергенције цильне функције на другом слоју са *AdaBoost* оптимизатором су приказане Сликом 6.18. Визуелизација стабилности оптимизатора је приказана Сликом 6.19.



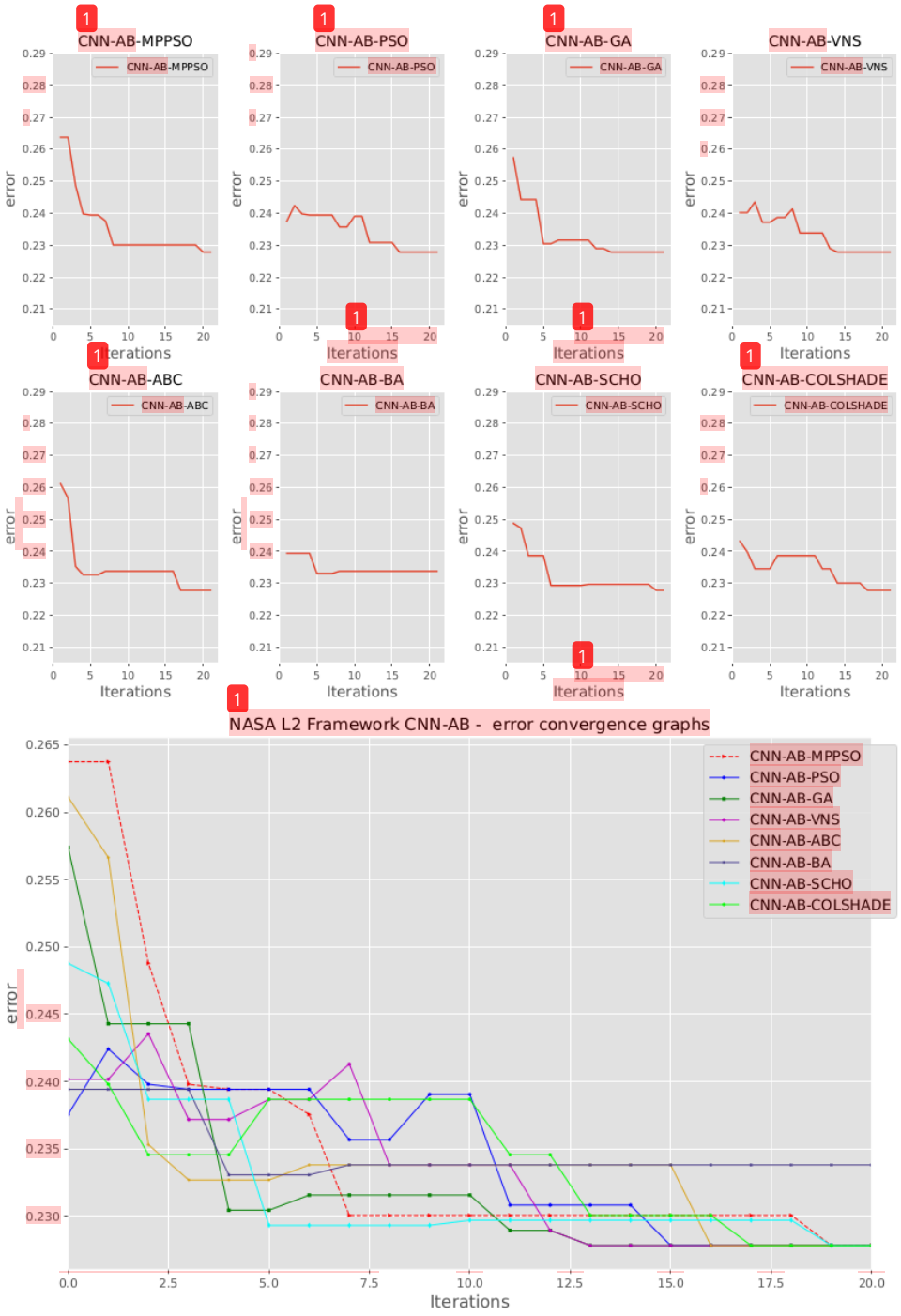
Слика 6.18: Конвергенција циљне функције *L2 Adaboost* модела.



Слика 6.19: Дистрибуције цильне и индикаторске функције *L2 AdaBoost* модела.

Предложени алгоритам показује одличне перформансе на приказаним графиконима, конвергирајући већ од 7. итерације након чега напредује до 19. итерације. Други оптимизатори проналазе солидна решења, али бржа конвергенција ограничава квалитет пронађеног решења.

Сличан исход су имали и експерименти индикаторске функције. Предложени алгоритам је поново дошао до најбољег решења до 19. итерације приказано Сликом 6.20.



Слика 6.20: Конвергенција индикаторске функције *L2 Adaboost* модела.

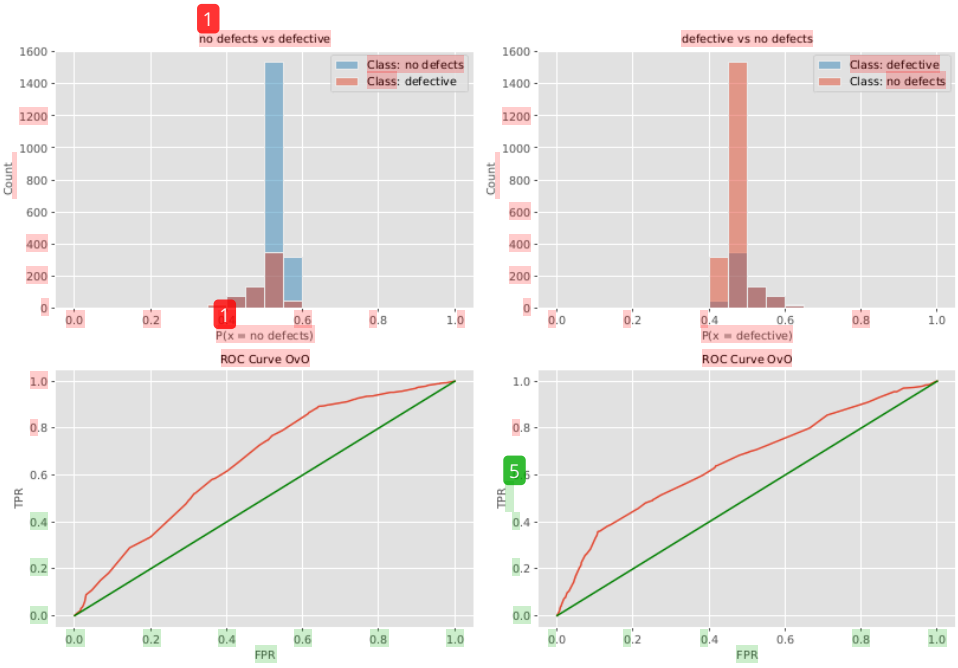
Најбољи *AdaBoost* модели по перформансама *L2* симулација сваког од тестираних оптимизатора су упоређени у Табели 6.23.

Табела 6.23: Детаљно поређење метрика модела са најбољим учинком *L2 AdaBoost* симулација.

Method	*metric*	*no error*	*error*	*accuracy*	*macro avg*	*weighted avg*
CNN-AB-MPPSO	*precision*	0.827586	**0.490909**	**0.772166**	**0.659248**	**0.751887**
	recall	**0.891892**	0.359401	**0.772166**	0.625646	**0.772166**
	f1-score	**0.858537**	0.414986	**0.772166**	**0.636761**	**0.758808**
CNN-AB-PSO	*precision*	0.827586	**0.490909**	**0.772166**	**0.659248**	**0.751887**
	recall	**0.891892**	0.359401	**0.772166**	0.625646	**0.772166**
	f1-score	**0.858537**	0.414986	**0.772166**	**0.636761**	**0.758808**
CNN-AB-GA	*precision*	0.827586	**0.490909**	**0.772166**	**0.659248**	**0.751887**
	recall	**0.891892**	0.359401	**0.772166**	0.625646	**0.772166**
	f1-score	**0.858537**	0.414986	**0.772166**	**0.636761**	**0.758808**
CNN-AB-VNS	*precision*	0.827586	**0.490909**	**0.772166**	**0.659248**	**0.751887**
	recall	**0.891892**	0.359401	**0.772166**	0.625646	**0.772166**
	f1-score	**0.858537**	0.414986	**0.772166**	**0.636761**	**0.758808**
CNN-AB-ABC	*precision*	0.827586	**0.490909**	**0.772166**	**0.659248**	**0.751887**
	recall	**0.891892**	0.359401	**0.772166**	0.625646	**0.772166**
	f1-score	**0.858537**	0.414986	**0.772166**	**0.636761**	**0.758808**
CNN-AB-BA	*precision*	**0.828416**	0.474468	0.766180	0.651442	0.748834
	recall	0.880792	**0.371048**	0.766180	**0.625920**	0.766180
	f1-score	0.853801	**0.416433**	0.766180	0.635117	0.755463
CNN-AB-SCHO	*precision*	0.827586	**0.490909**	**0.772166**	**0.659248**	**0.751887**
	recall	**0.891892**	0.359401	**0.772166**	0.625646	**0.772166**
	f1-score	**0.858537**	0.414986	**0.772166**	**0.636761**	**0.758808**
CNN-AB-COLSHADE	*precision*	0.827586	**0.490909**	**0.772166**	**0.659248**	**0.751887**
	recall	**0.891892**	0.359401	**0.772166**	0.625646	**0.772166**
	f1-score	**0.858537**	0.414986	**0.772166**	**0.636761**	**0.758808**
	support	2072	601			

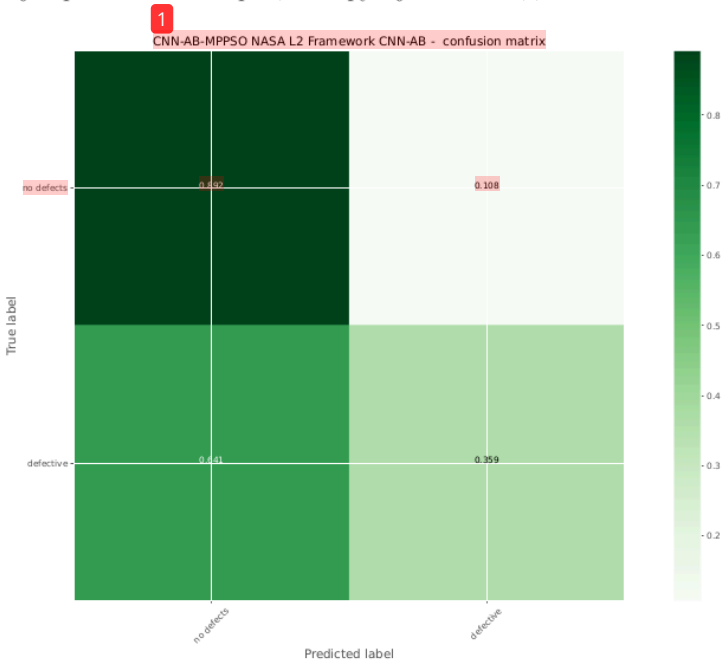
Неколико алгоритама употребљених у експериментима су остварили исте перформансе за неколико опажаних метрика. Ово указује да су ови оптимизатори приближно погодни за оптимизацију *AdaBoost* модела. Најбоља остварена тачност предикције је износила 0.772166, што је надмашило и најбољи резултат постигнут у *L1* симулацијама са употребом само *CNN*.

Додатне опсервације перформанси *AdaBoost L2* оптимизованог модела применом *MPPSO* алгоритма су доступне из графикона *ROC* криве приказаног Сликом 6.21.



Слика 6.21: *ROC* крива *L2 AdaBoost* оптимизованог модела.

Додатно је приказана и матрица конфузије истог модела Сликом 6.22.



Слика 6.22: Матрица конфузије *L2 AdaBoost* оптимизованог *L1* модела.

Параметри које су пронашли најбољи модели тестираних метахеуристика су приказани у Табели 6.24.

Табела 6.24: Селектовани параметри најбоље оптимизованих *L2 AdaBoost* модела.

Methods	*Number of estimators*	*Depth*	*Learning rate*
CNN-AB-MPPSO	17	1	0.388263
CNN-AB-PSO	17	1	0.395194
CNN-AB-GA	18	1	0.385504
CNN-AB-VNS	16	1	0.393270
CNN-AB-ABC	17	1	0.386973
CNN-AB-BA	37	2	0.100000
CNN-AB-SCHO	17	1	0.397827
CNN-AB-COLSHADE	16	1	0.389986

6.3.4 Симулације *XGBoost* класификације на другом слоју

Резултати експеримената *XGBoost L2* симулација су приложени у следећој Табели 6.25.

Табела 6.25: Исходи циљне функције *L2 XGBoost* модела.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
CNN-XG-MPPSO	**0.283956**	0.272256	0.277731	0.278157	0.003454	1.19E-05
CNN-XG-PSO	0.283057	**0.273192**	**0.279641**	**0.280789**	0.002805	7.87E-06
CNN-XG-GA	0.282818	0.271330	0.277524	0.277793	0.003237	1.05E-05
CNN-XG-VNS	0.282944	0.272961	0.278091	0.278468	**0.002479**	**6.14E-06**
CNN-XG-ABC	0.282813	0.264134	0.271095	0.269931	0.004398	1.93E-05
CNN-XG-BA	0.282907	0.258011	0.270776	0.271714	0.006936	4.81E-05
CNN-XG-SCHO	0.281937	0.270017	0.277780	0.278442	0.002853	8.14E-06
CNN-XG-COLSHADE	0.281011	0.271270	0.277472	0.278075	0.002513	6.32E-06

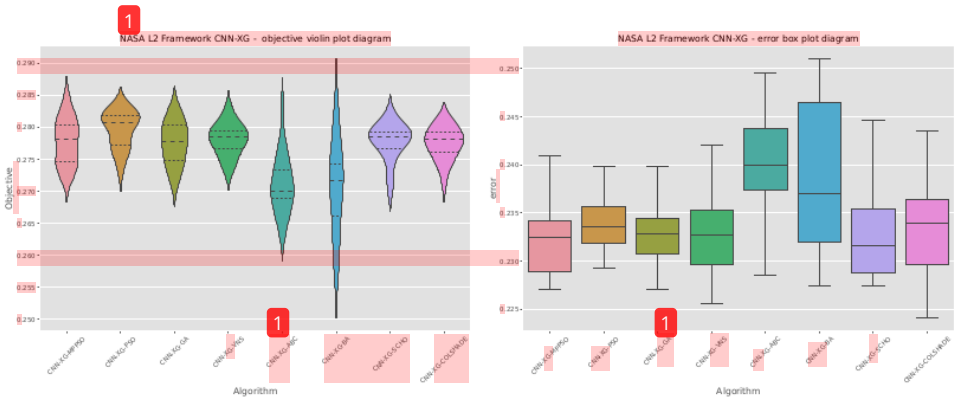
Предложена метода постиже најбољи резултат за метрику најбољег резултата циљне функције са вредношћу од 0.283956. По погледу других измерених метрика као што су медијана, средња и најгора вредност, *PSO* алгоритам показује најбоље перформансе. Најбољу стабилност показао је *VNS*. Резултати *L2* експеримената за индикаторску функцију су приказани у Табели 6.26.

Табела 6.26: Исходи индикаторске функције *L2 XGBoost* модела.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
CNN-XG-MPPSO	0.228956	0.232697	**0.232043**	0.232510	0.003383	1.14E-05
CNN-XG-PSO	0.233820	0.234194	0.233876	0.233633	**0.002654**	**7.04E-06**
CNN-XG-GA	0.228582	0.234194	0.232772	0.232884	0.003501	1.23E-05
CNN-XG-VNS	**0.225589**	0.235316	0.232866	0.232697	0.004007	1.61E-05
CNN-XG-ABC	0.239805	0.243172	0.240591	0.239993	0.004909	2.41E-05
CNN-XG-BA	0.231949	0.245043	0.239263	0.237000	0.007681	5.90E-05
CNN-XG-SCHO	0.233446	0.240928	0.232941	**0.231575**	0.005156	2.66E-05
CNN-XG-COLSHADE	0.227086	**0.226712**	0.233651	0.234007	0.004904	2.41E-05

Осврtом на резултате из претходне табеле могуће је приметити да је предложено решење имало најбољи резултат за средњу вредност. Експерименти са индикаторском функцијом такође доприносе подржавању *NFL* теореме. Најбољу стабилност је постигао *PSO* алгоритам.

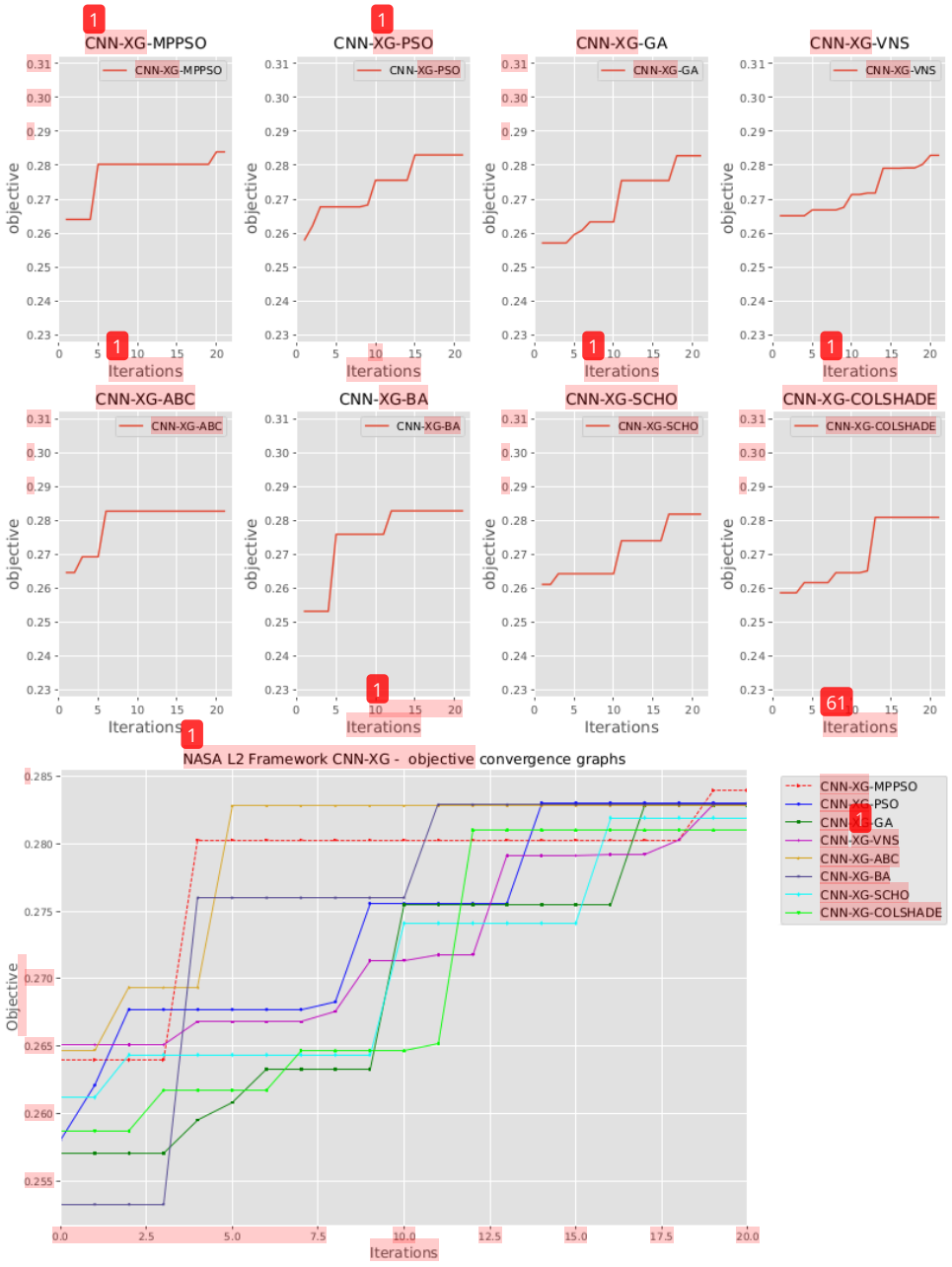
Визуелизација стабилности оптимизатора је приказа Сликом 6.23.



Слика 6.23: Дистрибуције цильне и индикаторске функције *L2 XGBoost* модела.

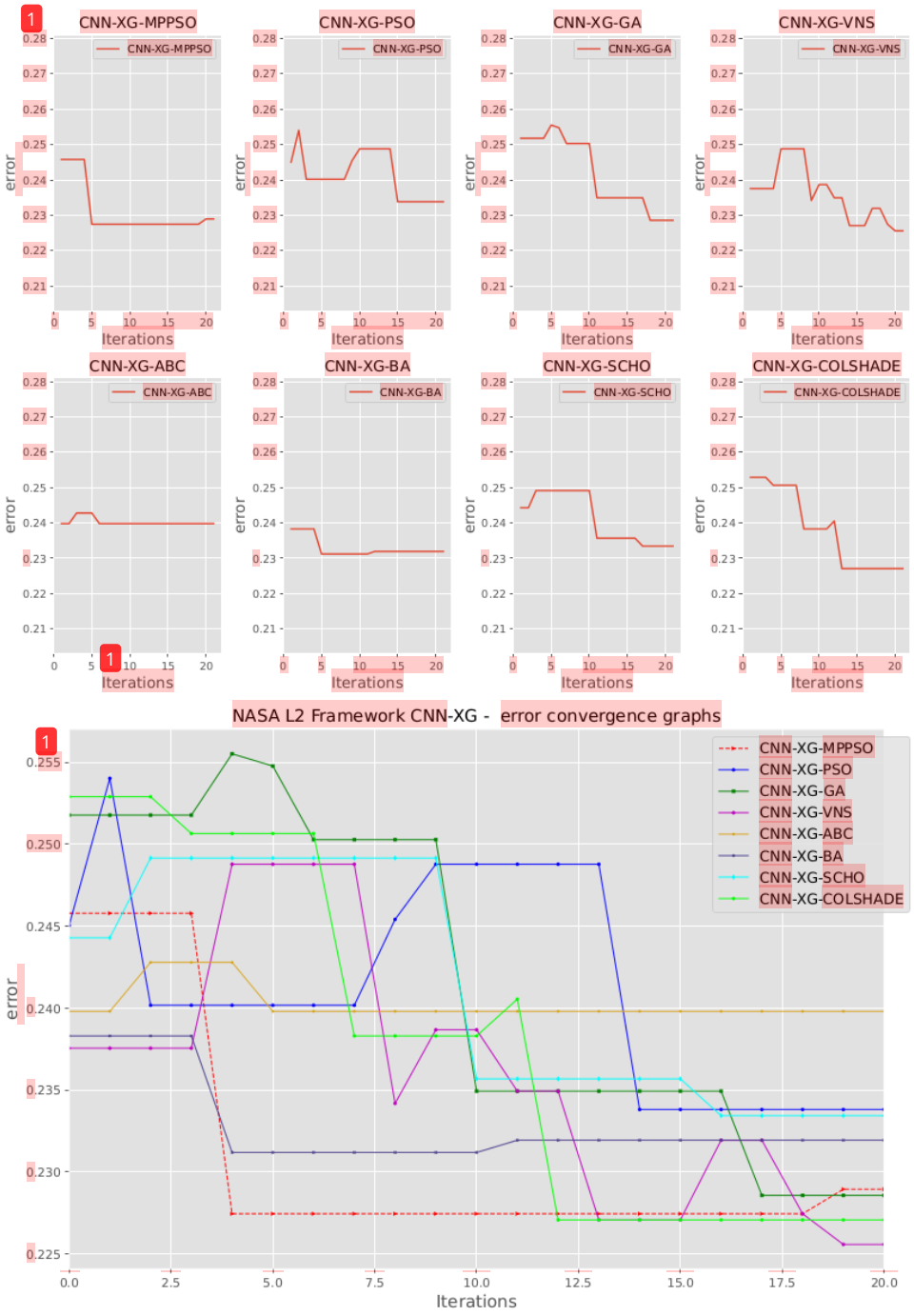
Насупрот томе што предложена метахеуристика остварује најбоље перформансе по питању цильне функције, основна верзија *PSO* алгоритма остварује бољу групу резултата у простору претраге. Лоши резултати по питању стабилности су забележени код *BA* оптимизатора, а *ABC* се фокусирао на субоптималне регије у вишеструким инстанцама.

Резултати оптимизације *L2 XGBoost* цильне функције осликавајући стопе конвергенције најбољих модела су приказани Сликом 6.24.



Слика 6.24: Конвергенција цилне функције *L2 XGBoost* модела.

Резултати оптимизације *L2 XGBoost* индикаторске функције осликавајући стопе конвергенције најбољих модела су приказани Сликом 6.25.



Слика 6.25: Конвергенција индикаторске функције *L2 XGBoost* модела.

Предложени алгоритам постиже сличне резултате и проналази обећавајуће регионе простора претраге. Остали алгоритми су се показали лошије од предложеног решења, али зато ово решење постиже слабије резултате индикаторске функције. Чест случај је да се перформансе једне метрике не осликавају на другу метрику због *NFL* теореме. Додатно треба узети у обзир да циљ оптимизације предложеног алгоритма није била индикаторска функција, али да алгоритам свакако постиже задовољавајуће перформансе.

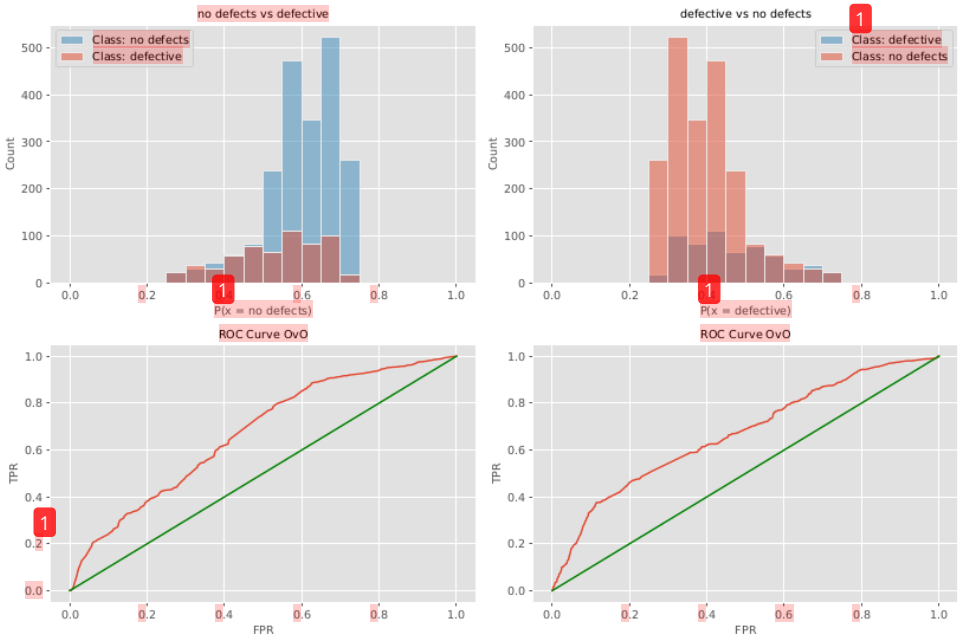
Поређења најбољих модела *L2 XGBoost* модела оптимизованих употребом сваког од алгоритама су детаљније приказана у Табели 6.27.

Табела 6.27: Детаљно поређење метрика модела са најбољим учинком *L2 XGBoost* симулација.

Method	*metric*	*no error*	*error*	*accuracy*	*macro avg*	*weighted avg*
CNN-XG-MPPSO	*precision*	0.830018	0.488069	0.771044	0.659044	0.753134
	recall	0.886100	0.374376	0.771044	0.630238	0.771044
	f1-score	0.857143	0.423729	0.771044	**0.640436**	0.759694
CNN-XG-PSO	*precision*	**0.831729**	0.475610	0.766180	0.653669	0.751658
	recall	0.875483	**0.389351**	0.766180	**0.632417**	0.766180
	f1-score	0.853045	**0.428179**	0.766180	0.640612	0.757518
CNN-XG-GA	*precision*	0.829499	0.489035	0.771418	0.659267	0.752949
	recall	0.887548	0.371048	0.771418	0.629298	0.771418
	f1-score	0.857543	0.421949	0.771418	0.639746	0.759603
CNN-XG-VNS	*precision*	0.828342	**0.497706**	**0.774411**	**0.663024**	**0.754001**
	recall	**0.894305**	0.361065	**0.774411**	0.627685	**0.774411**
	f1-score	**0.860060**	0.418515	**0.774411**	0.639288	**0.760783**
CNN-XG-ABC	*precision*	0.772056	0.209091	0.679386	0.490573	0.645478
	recall	0.832046	0.153078	0.679386	0.492562	0.679386
	f1-score	0.800929	0.176753	0.679386	0.488841	0.660589
CNN-XG-BA	*precision*	0.830902	0.480167	0.768051	0.655535	0.752043
	recall	0.879826	0.382696	0.768051	0.631261	0.768051
	f1-score	0.854665	0.425926	0.768051	0.640295	0.758267
CNN-XG-SCHO	*precision*	0.831199	0.476386	0.766554	0.653792	0.751422
	recall	0.876931	0.386023	0.766554	0.631477	0.766554
	f1-score	0.853452	0.426471	0.766554	0.639961	0.757449
CNN-XG-COLSHADE	*precision*	0.828328	0.493213	0.772914	0.660770	0.752980
	recall	0.891892	0.362729	0.772914	0.627310	0.772914
	f1-score	0.858936	0.418025	0.772914	0.638480	0.759801
	support	2072	601			

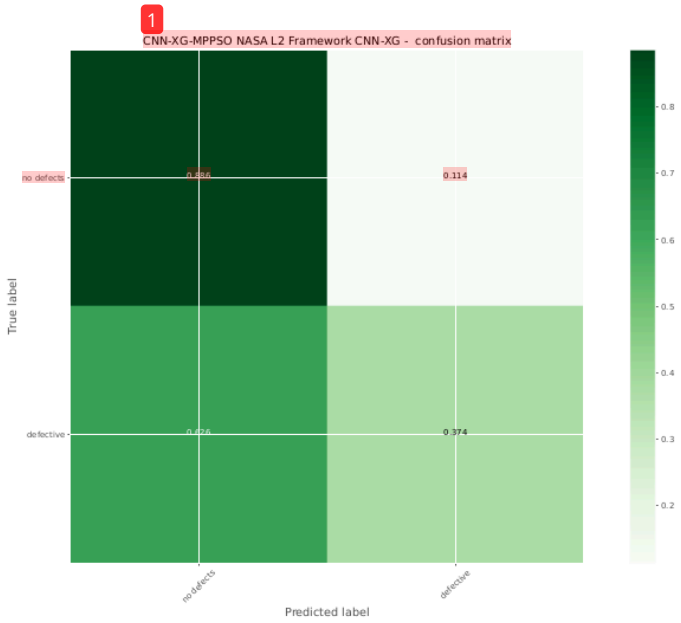
Узимајући у обзир природу проблема, *VNS* алгоритам има јасну предност у односу на остала решења, што је још један аргумент који додатно потврђује *NFL* теорему. Нажалост ова чињеница не доприноси решењу овог проблема, већ га само потврђује. Експериментацијом, односно емпиријски је потребно закључити које решење је најбоље за који тип проблема и не само тип већ и конкретан случај проблема. Додатно треба нагласити да је неопходно узети у обзир све анализиране метрике при одлучивању који је најбољи алгоритам за тражени проблем оптимизације.

ROC крива *XGBoost L2* модела оптимизованог са *MPPSO* алгоритмом је приказана Сликом 6.26.



Слика 6.26: *ROC* крива *L2 XGBoost* оптимизованог *L1* модела.

Матрица конфузије *XGBoost L2* модела оптимизованог са *MPPSO* алгоритмом је приказана Сликом 6.27.



Слика 6.27: Матрица конфузије *L2 XGBoost* оптимизованог *L1* модела.

Резултати узастопних симулација и то за најбољи скуп хиперпараметара који је сваки модел остварио су приказани у Табели 6.28.

Табела 6.28: Селектовани параметри најбоље оптимизованих *L2 XGBoost* модела.

Method	*Learning Rate*	*max_child_weight*	*subsample*	*collsample_bytree*	*max_depth*	*gamma*
CNN-XG-MPPSO	0.100000	10.000000	0.983197	0.293945	3	0.713646
CNN-XG-PSO	0.100000	2.357106	0.968264	0.363649	5	0.015409
CNN-XG-GA	0.107403	1.691592	0.950345	0.319984	3	0.000000
CNN-XG-VNS	0.175166	4.645308	1.000000	0.284932	4	0.757203
CNN-XG-ABC	0.285948	6.211737	0.744753	0.678273	6	0.779172
CNN-XG-BA	0.206396	3.716090	1.000000	0.322232	4	0.528585
CNN-XG-SCHO	0.100000	3.353770	1.000000	0.269461	3	0.000000
CNN-XG-COLSHADE	0.100000	2.409330	0.989736	0.306208	3	0.050836

6.3.5 Симулације *AdaBoost* класификације са *NLP* метода на другом слоју

Резултати оптимизације *AdaBoost NLP* класификатора за циљну функцију су приложени у Табели 6.29.

Табела 6.29: Исходи циљне функције *NLP L2 AdaBoost* модела.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
AB-MPPSO	**0.959947**	0.955168	0.956996	0.956397	0.001601	2.56E-06
AB-PSO	0.959207	**0.957205**	**0.958233**	0.957872	**0.000780**	**6.08E-07**
AB-GA	0.959780	0.955059	0.957057	0.957037	0.001547	2.39E-06
AB-VNS	0.956367	0.953758	0.955049	0.955026	0.000913	8.34E-07
AB-ABC	0.957133	0.952387	0.955170	0.955638	0.001554	2.42E-06
AB-BA	**0.959947**	0.950244	0.956981	**0.959207**	0.003716	1.38E-05
AB-SCHO	0.957769	0.954325	0.955999	0.955834	0.001109	1.23E-06
AB-COLSHADE	0.958470	0.954542	0.956181	0.956367	0.001371	1.88E-06

Предложени алгоритам показује најбоље резултате за најбољу метрику, са резултатом циљне функције од 0.959947. Исти резултат је постигао и *BA*. Основна верзија *PSO* алгоритма се још увек показује као најстабилнији алгоритам доминирајући категорију средње вредности, док је *BA* показао најбољи резултат за медијану.

Резултати класификације са *AdaBoost* и *NLP* метода су приказани у Табели 6.30.

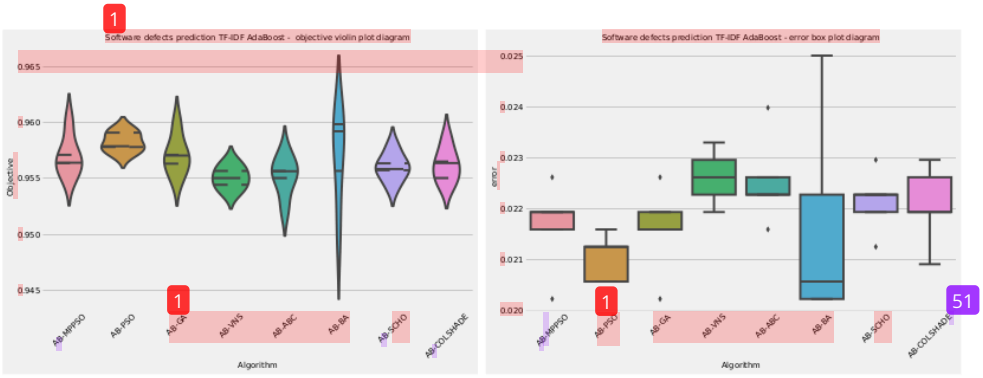
Табела 6.30: Исходи индикаторске функције *NLP L2 AdaBoost* модела.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
AB-MPPSO	**0.020219**	0.022618	0.021659	0.021933	0.000793	6.29E-07
AB-PSO	0.020562	**0.021590**	**0.021042**	0.021247	**0.000411**	**1.69E-07**
AB-GA	**0.020219**	0.022618	0.021590	0.021590	0.000781	6.11E-07
AB-VNS	0.021933	0.023304	0.022618	0.022618	0.000485	2.35E-07
AB-ABC	0.021590	0.023989	0.022550	0.022276	0.000793	6.29E-07
AB-BA	**0.020219**	0.025017	0.021659	**0.020562**	0.001844	3.40E-06
AB-SCHO	0.021247	0.022961	0.022138	0.022276	0.000557	3.10E-07
AB-COLSHADE	0.020905	0.022961	0.022070	0.021933	0.000706	4.98E-07

Високу стабилност показује оригинални *PSO* али и *VNS* такође показује обећавајуће резултате. Приметити да је *PSO* остварио најбоље резултате за најгору метрику

и средњу вредност, док предложено решење постиже најбољи резултат.

Визуелна поређења стабилности алгоритама су приказана Сликом 6.28.



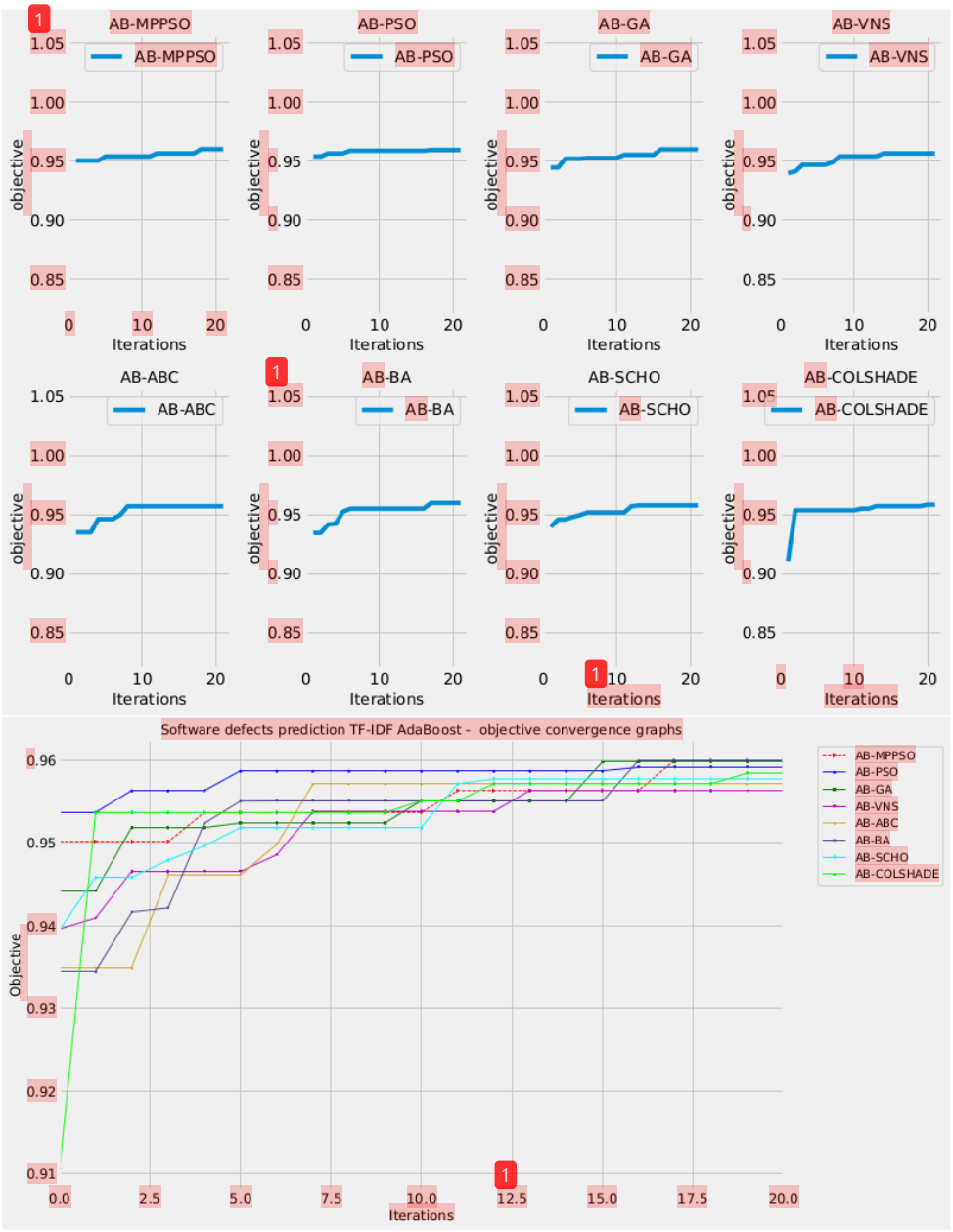
Слика 6.28: Дистрибуција циљне и индикаторске функције *NLP L2 Adaboost* модела.

Приказана слика потврђује стабилност *PSO*, али најбољу групу решења ипак постиже његова унапређена верзија.

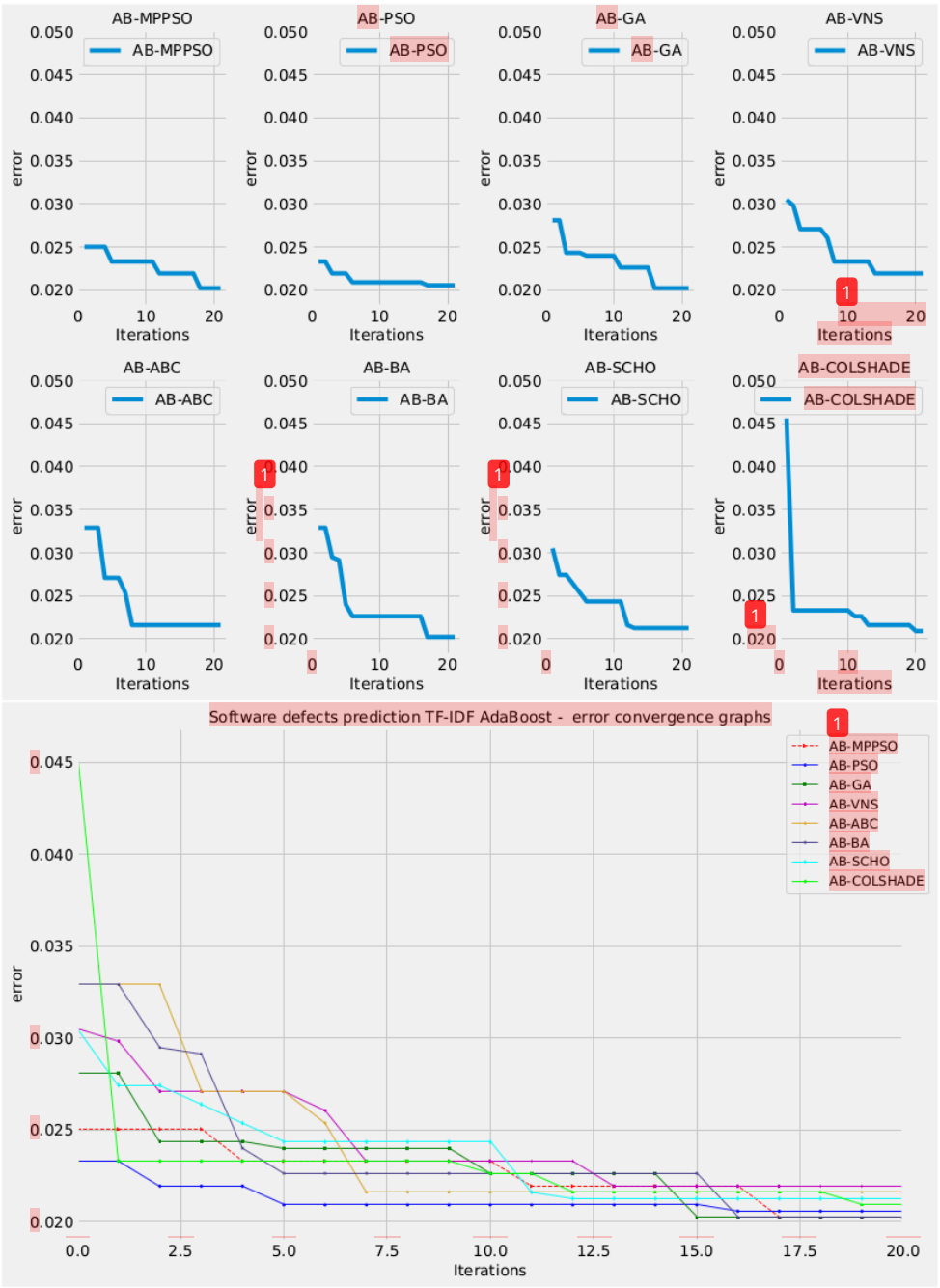
Стопе конвергенције за циљну функцију оптимизације са *Adaboost* и *NLP* методама су приказане Сликом 6.29.

Већина алгоритама постиже добре резултате конвергенције, али одређени имају проблема са заглављивањем у субоптималним регионима. Предложени алгоритам је најбоље решење пронашао у 17. итерацији. Стопе конвергенције за индикаторску функцију оптимизације са *Adaboost* и *NLP* методама су приказане Сликом 6.30. Слична опажања су могућа и са слике која описује стопе конвергенције алгоритама за индикаторску функцију.

Поређења најбољих модела *AdaBoost NLP* модела оптимизованих употребом сваког од алгоритама су детаљније приказана у Табели 6.31.



Слика 6.29: Конвергенција циљне функције *NLP L2 AdaBoost* модела.



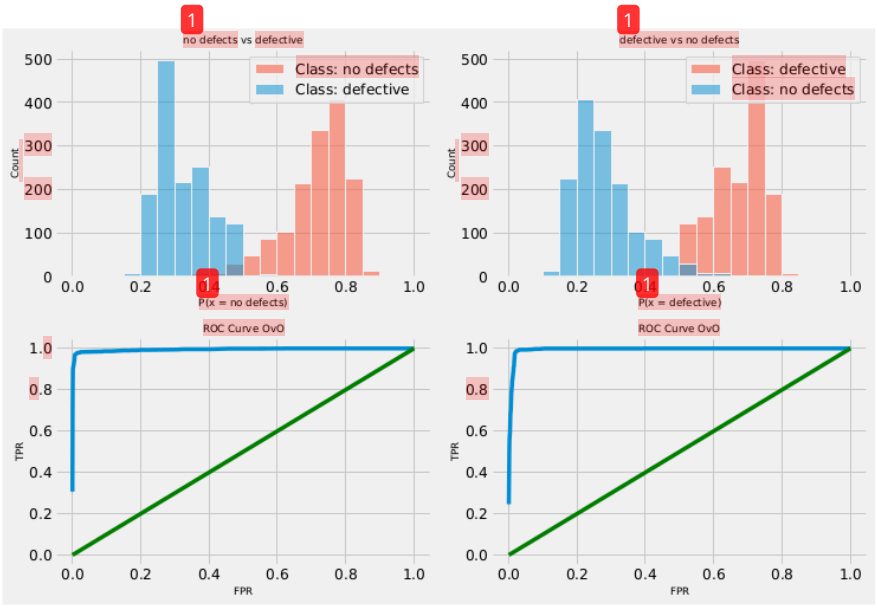
Слика 6.30: Конвергенција индикаторске функције *NLP L2 AdaBoost* модела.

Табела 6.31: Детаљно поређење метрика модела са најбољим учинком *NLP L2 AdaBoost* симулација.

Method	*metric*	*no error*	*error*	*accuracy*	*macro avg*	*weighted avg*
AB-MPPSO	*precision*	0.993776	0.966033	**0.979781**	**0.979904**	**0.980170**
	recall	0.966375	**0.993711**	**0.979781**	**0.980043**	**0.979781**
	f1-score	0.979884	**0.979676**	**0.979781**	**0.979780**	0.979782
AB-PSO	*precision*	0.992409	0.966644	0.979438	0.979526	0.979773
	recall	0.967048	0.992313	0.979438	0.979680	0.979438
	f1-score	**0.979564**	0.979310	0.979438	0.979437	0.979440
AB-GA	*precision*	0.990385	**0.969220**	**0.979781**	0.979802	0.980006
	recall	**0.969738**	0.990217	**0.979781**	0.979977	**0.979781**
	f1-score	0.979952	0.979606	**0.979781**	0.979779	**0.979783**
AB-VNS	*precision*	0.989003	0.967191	0.978067	0.978097	0.978306
	recall	0.967720	0.988819	0.978067	0.978270	0.978067
	f1-score	0.978246	0.977885	0.978067	0.978066	0.978069
AB-ABC	*precision*	0.991034	0.965940	0.978410	0.978487	0.978728
	recall	0.966375	0.990915	0.978410	0.978645	0.978410
	f1-score	0.978550	0.978268	0.978410	0.978409	0.978412
AB-BA	*precision*	0.993776	0.966033	**0.979781**	**0.979904**	**0.980170**
	recall	0.966375	**0.993711**	**0.979781**	**0.980043**	**0.979781**
	f1-score	0.979884	**0.979676**	**0.979781**	**0.979780**	0.979782
AB-SCHO	*precision*	0.990365	0.967235	0.978753	0.978800	0.979022
	recall	0.967720	0.990217	0.978753	0.978968	0.978753
	f1-score	0.978912	0.978591	0.978753	0.978751	0.978754
AB-COLSHADE	*precision*	0.991047	0.967258	0.979095	0.979152	0.979381
	recall	0.967720	0.990915	0.979095	0.979318	0.979095
	f1-score	0.979245	0.978944	0.979095	0.979094	0.979097
	support	1487	1431			

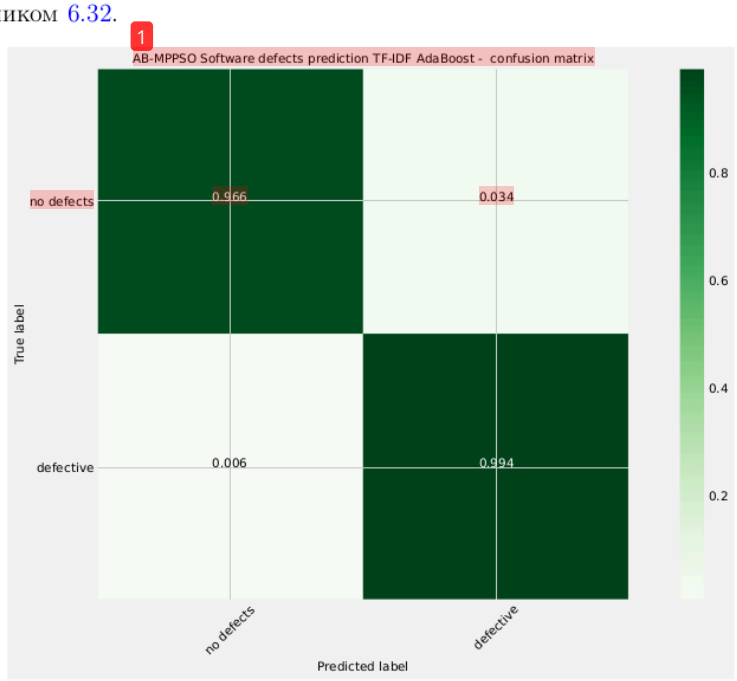
Добре резултате постижу све методе што потврђује да *NLP* методе унапређују коначни модел за проблем детекције грешке.

ROC криву *AdaBoost NLP* модела оптимизованог помоћу *MPPSO* приказује Слика 6.31.



Слика 6.31: *ROC* крива *NLP L2 AdaBoost* оптимизованог *L1* модела.

Матрица конфузије *AdaBoost NLP* модела оптимизованог помоћу *MPPSO* је приказана Сликом 6.32.



Слика 6.32: Матрица конфузије *NLP L2 AdaBoost* оптимизованог *L1* модела.

Напослетку су приказани резултати узастопних симулација и то за најбољи скуп

хиперпараметара који је сваки модел остварио у Табели 6.32.

Табела 6.32: Селектовани параметри најбоље оптимизованих *NLP L2 AdaBoost* модела.

Methods	*Number of estimators*	*Depth*	*Learning rate*
AB-MPPSO	15	5	1.425870
AB-PSO	15	6	1.806671
AB-GA	14	6	1.657350
AB-VNS	15	6	1.357072
AB-ABC	14	6	1.761995
AB-BA	15	6	1.808268
AB-SCHO	14	6	1.669562
AB-COLSHADE	13	6	1.464666

6.3.6 Симулације *XGBoost* класификације са *NLP* методама на другом слоју

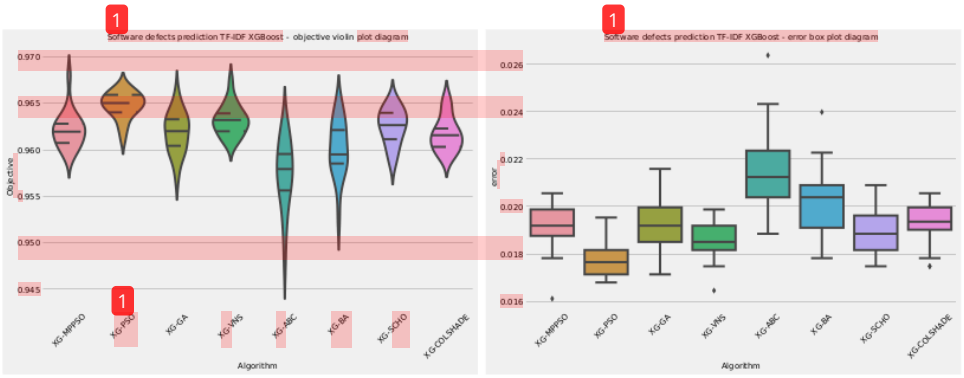
Резултати класификације *XGBoost* са применом *NLP* метода су приказани у Табели 6.33.

Табела 6.33: Исходи циљне функције *NLP L2 XGBoost* модела.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
XG-MPPSO	**0.968036**	0.959172	0.962128	0.961967	0.001908	3.64E-06
XG-PSO	0.966730	**0.961151**	**0.964797**	**0.965050**	**0.001395**	**1.95E-06**
XG-GA	0.966098	0.957067	0.961848	0.962024	0.002147	4.61E-06
XG-VNS	0.967399	0.960652	0.963171	0.963222	0.001609	2.59E-06
XG-ABC	0.962689	0.948075	0.957429	0.957948	0.003626	1.31E-05
XG-BA	0.964691	0.952595	0.960146	0.959524	0.002959	8.76E-06
XG-SCHO	0.965394	0.958438	0.962483	0.962671	0.001889	3.57E-06
XG-COLSHADE	0.965394	0.959140	0.961709	0.961590	0.001811	3.28E-06

Предложена метахеуристика постиже најбоље резултате за најбољу метрику. Не треба занемарити да је и оригинална верзија предложеног алгоритма постигла одличне резултате у свим категоријама као и да још једном има највећу стабилност.

Резултати оптимизације *XGBoost NLP* модела за индикаторску функцију су приказани у Табели 6.34.



Слика 6.33: Дистрибуција циљне и индикаторске функције *NLP L2 XGBoost* модела.

Табела 6.34: Исходи индикаторске функције *NLP L2 XGBoost* модела.

Method	*Best*	*Worst*	*Mean*	*Median*	*Std*	*Var*
XG-MPPSO	**0.016107**	0.020562	0.019106	0.019191	0.000969	9.38E-07
XG-PSO	0.016792	**0.019534**	**0.017752**	**0.017649**	**0.000699**	**4.89E-07**
XG-GA	0.017135	0.021590	0.019243	0.019191	0.001074	1.15E-06
XG-VNS	0.016450	0.019877	0.018574	0.018506	0.000815	6.65E-07
XG-ABC	0.018849	0.026388	0.021470	0.021247	0.001856	3.44E-06
XG-BA	0.017820	0.023989	0.020099	0.020391	0.001503	2.26E-06
XG-SCHO	0.017478	0.020905	0.018934	0.018849	0.000950	9.03E-07
XG-COLSHADE	0.017478	0.020562	0.019311	0.019363	0.000921	8.49E-07

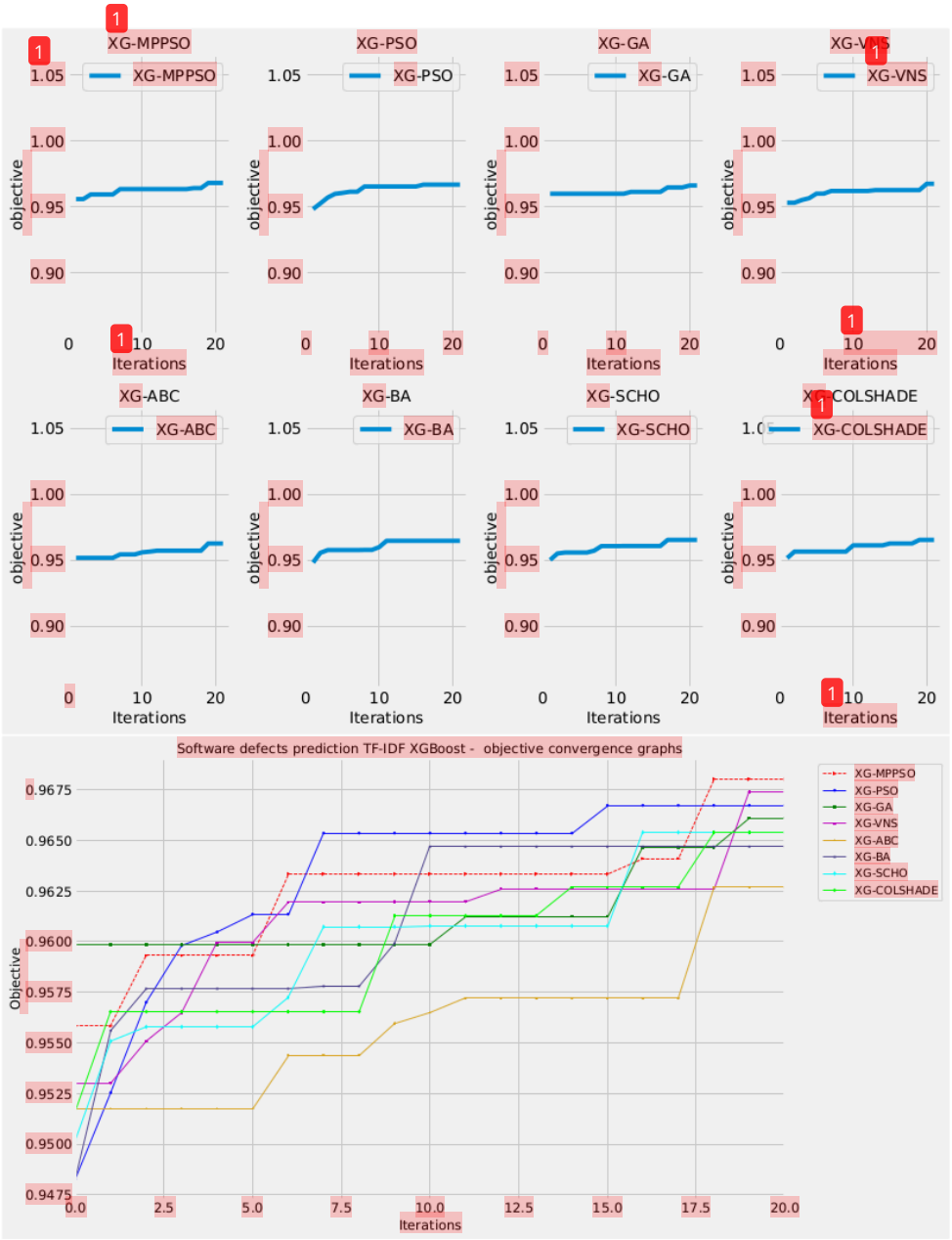
Поређење стабилности најбољих *XGBoost NLP* модела је визуелизовано Сликом 6.33.

Резултати поново указују на исто, предложена метрика је доминантна у постизању најоптималнијих резултата, док оригинални аглоритам има најбољу стабилност.

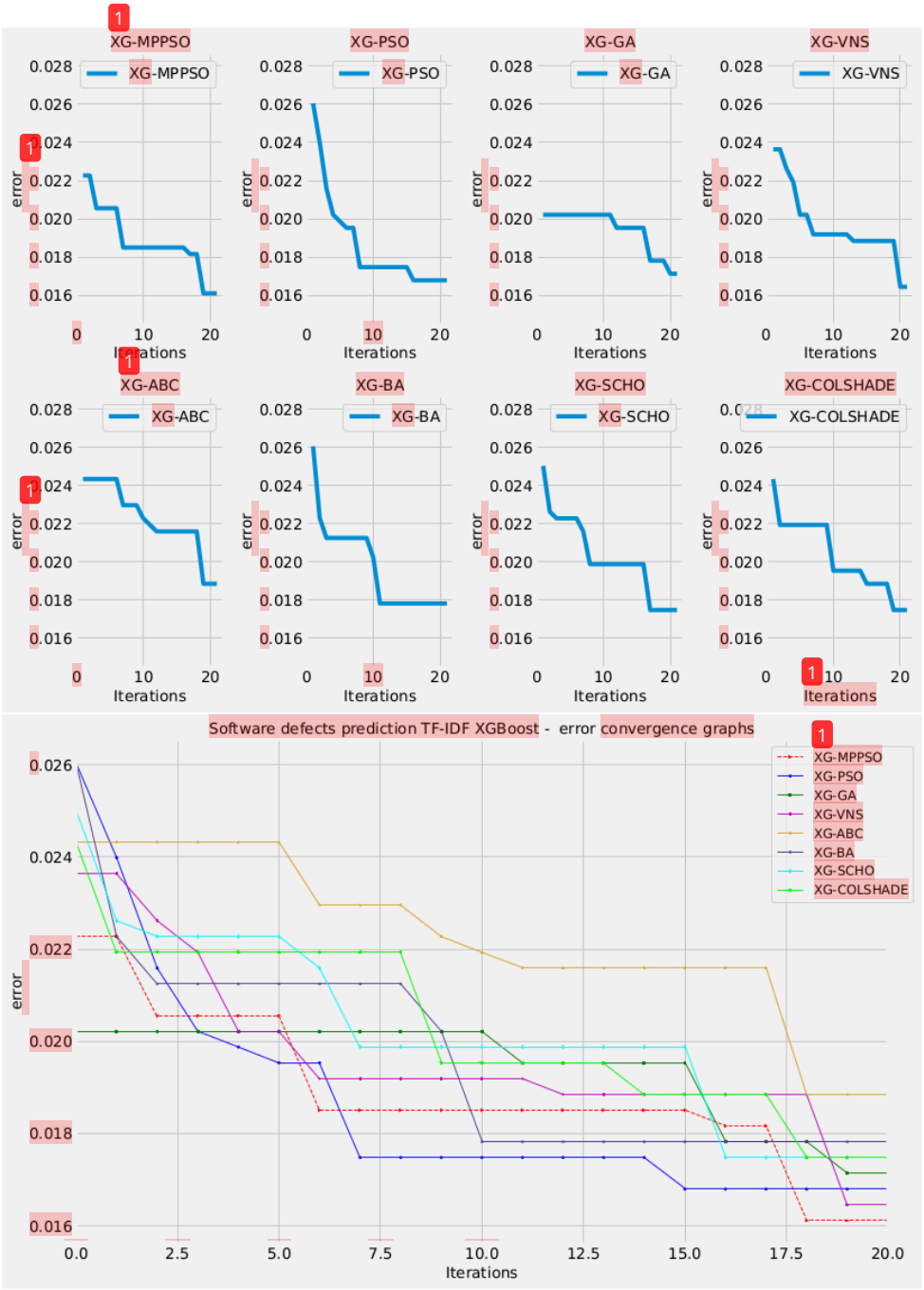
Стопе конвергенције за циљну функцију оптимизације са *XGBoost* и *NLP* методама су приказане Сликом 6.34.

Предложена метахеуристика превазилази локалне оптимуме и достиже најбоље решење до 19. итерације. Из наведеног се може закључити да је постигнуто унапређење по питању експлорације у каснијим итерацијама где оригинални *PSO* посустаје. Стопе конвергенције за индикаторску функцију оптимизације са *XGBoost* и *NLP* методама су приказане Сликом 6.35. Слични закључци су доступни и из визуелизације стопа конвергенције модела за индикаторску функцију.

Поређења најбољих модела *XGBoost NLP* модела оптимизованих употребом сваког од алгоритама су детаљније приказана у Табели 6.35.



Слика 6.34: Конвергенција циљне функције *NLP L2 XGBoost* модела.



Слика 6.35: Конвергенција индикаторске функције *NLP L2 XGBoost* модела.

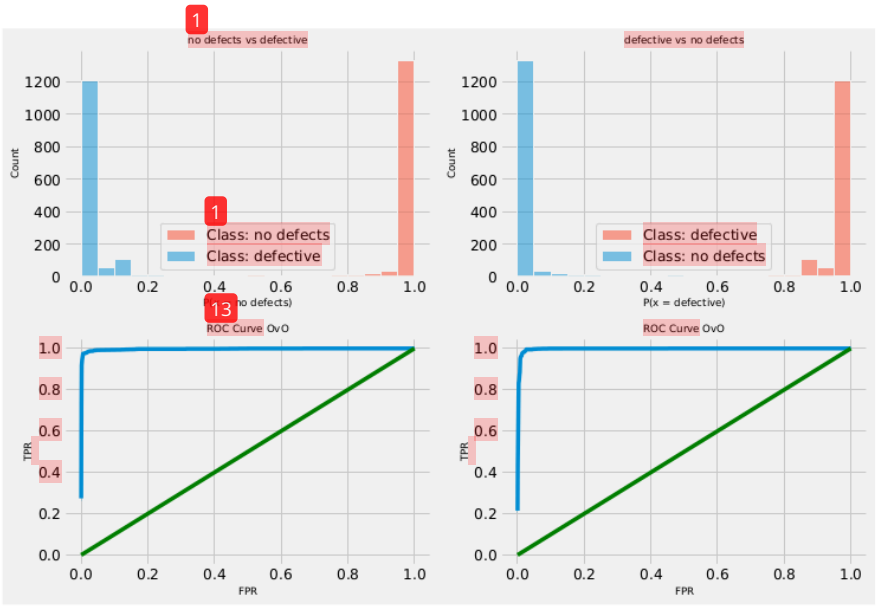
Табела 6.35: Детаљно поређење метрика модела са најбољим учинком *NLP L2 XGBoost* симулација.

Method	*metric*	*no error*	*error*	*accuracy*	*macro avg*	*weighted avg*
XG-MPPSO	*precision*	0.995186	**0.972678**	**0.983893**	**0.983932**	**0.984148**
	recall	**0.973100**	0.995108	**0.983893**	**0.984104**	**0.983893**
	f1-score	**0.984019**	**0.983765**	**0.983893**	**0.983892**	**0.983895**
XG-PSO	*precision*	0.995862	0.970708	0.983208	0.983285	0.983527
	recall	0.971083	0.995807	0.983208	0.983445	0.983208
	f1-score	0.983316	0.983098	0.983208	0.983207	0.983209
XG-GA	*precision*	**0.996545**	0.969409	0.982865	0.982977	0.983237
	recall	0.969738	**0.996506**	0.982865	0.983122	0.982865
	f1-score	0.982958	0.982771	0.982865	0.982864	0.982866
XG-VNS	*precision*	0.995865	0.971370	0.983550	0.983618	0.983853
	recall	0.971755	0.995807	0.983550	0.983781	0.983550
	f1-score	0.983662	0.983437	0.983550	0.983550	0.983552
XG-ABC	*precision*	0.995159	0.967391	0.981151	0.981275	0.981542
	recall	0.967720	0.995108	0.981151	0.981414	0.981151
	f1-score	0.981248	0.981054	0.981151	0.981151	0.981153
XG-BA	*precision*	0.995169	0.969367	0.982180	0.982268	0.982516
	recall	0.969738	0.995108	0.982180	0.982423	0.982180
	f1-score	0.982289	0.982069	0.982180	0.982179	0.982181
XG-SCHO	*precision*	0.995856	0.969388	0.982522	0.982622	0.982876
	recall	0.969738	0.995807	0.982522	0.982772	0.982522
	f1-score	0.982624	0.982420	0.982522	0.982522	0.982524
XG-COLSHADE	*precision*	0.995856	0.969388	0.982522	0.982622	0.982876
	recall	0.969738	0.995807	0.982522	0.982772	0.982522
	f1-score	0.982624	0.982420	0.982522	0.982522	0.982524
	support	1487	1431			

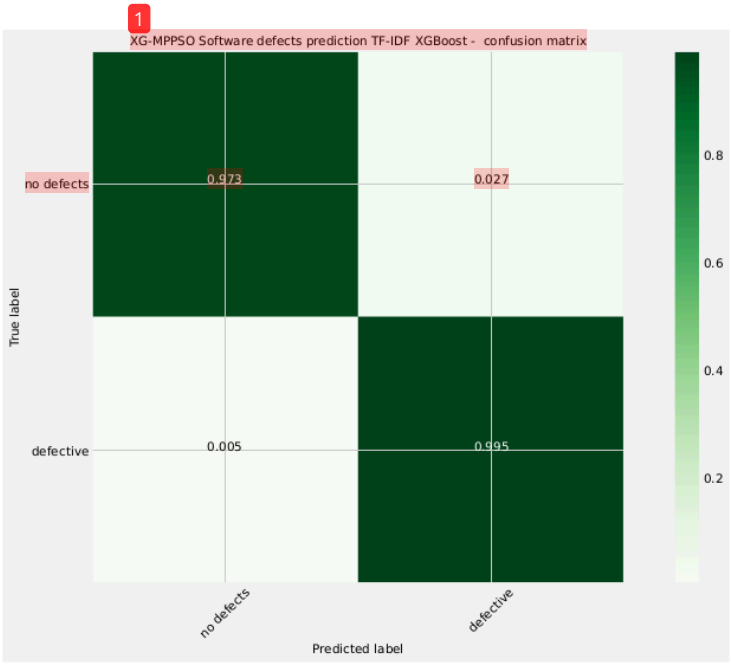
Најбољи резултат је постигнут од стране предложене метахеуристике и то 0.983893.

ROC крива *XGBoost NLP* модела оптимизованог помоћу *MPPSO* је приказана Сликом 6.36.

Матрица конфузије *XGBoost NLP* модела оптимизованог помоћу *MPPSO* је приказана Сликом 6.37.



Слика 6.36: *ROC* крива *NLP L2 XGBoost* оптимизованог *L1* модела.



Слика 6.37: Матрица конфузије *NLP L2 XGBoost* оптимизованог *L1* модела.

Напослетку су приказани резултати узастопних симулација и то за најбољи скуп хиперпараметара који је сваки модел остварио у Табели 6.36.

Табела 6.36: Селектовани параметри најбоље оптимизованих *NLP L2 XGBoost* модела.

Method	*Learning Rate*	*max_child_weight*	*subsample*	*collsample_bytree*	*max_depth*	*gamma*
XG-MPPSO	0.878864	1.000000	0.886550	0.447144	10	0.001162
XG-PSO	0.900000	1.000000	1.000000	0.418138	10	0.137715
XG-GA	0.900000	1.000000	1.000000	0.380366	10	0.000000
XG-VNS	0.874556	1.000000	1.000000	0.389317	10	0.800000
XG-ABC	0.900000	1.000000	0.920135	0.412610	7	0.512952
XG-BA	0.900000	1.191561	1.000000	0.451404	10	0.800000
XG-SCHO	0.900000	1.000000	1.000000	0.425699	10	0.161777
XG-COLSHADE	0.900000	1.000000	1.000000	0.369307	10	0.262928

6.3.7 Анализа

Резултати експеримената извршених на HACA скупу података који представљају основне моделе без оптимизације су приказани у Табели 6.37.

Табела 6.37: Поређење са резултатима основних модела за HACA експеримент.

Models	Accuracy	MCC
CNN-MPPSO	76.88%	0.273280
CNN-AB-MPPSO	77.22%	0.282906
CNN-XG-MPPSO	77.10%	0.283956
AdaBoost	73.58%	0.245921
XGBoost	73.21%	0.241493
CatBoost	72.86%	0.238149
Random forest	75.54%	0.255372
SVM	74.37%	0.247345
DNN 2 hidden layers	76.86%	0.266794
DNN 3 hidden layers	76.72%	0.266037
DNN 3 hidden layers (dropout 0.1)	77.05%	0.268044

Резултати експеримената извршених на *NLP* скупу података који представљају основне моделе без оптимизације су приказани у Табели 6.38.

Табела 6.38: Поређење са резултатима основних модела за *NLP* експеримент.

Models	Accuracy	MCC
AB-MPPSO	97.98%	0.959947
XG-MPPSO	98.38%	0.968036
AdaBoost	95.38%	0.932849
XGBoost	95.85%	0.934185
CatBoost	95.77%	0.933941
Random forest	94.62%	0.924924
SVM	93.39%	0.914582
DNN 2 hidden layer	97.91%	0.943741
DNN 3 hidden layers	97.83%	0.948537
DNN3 hidden layers (dropout 0.1)	98.14%	0.946093

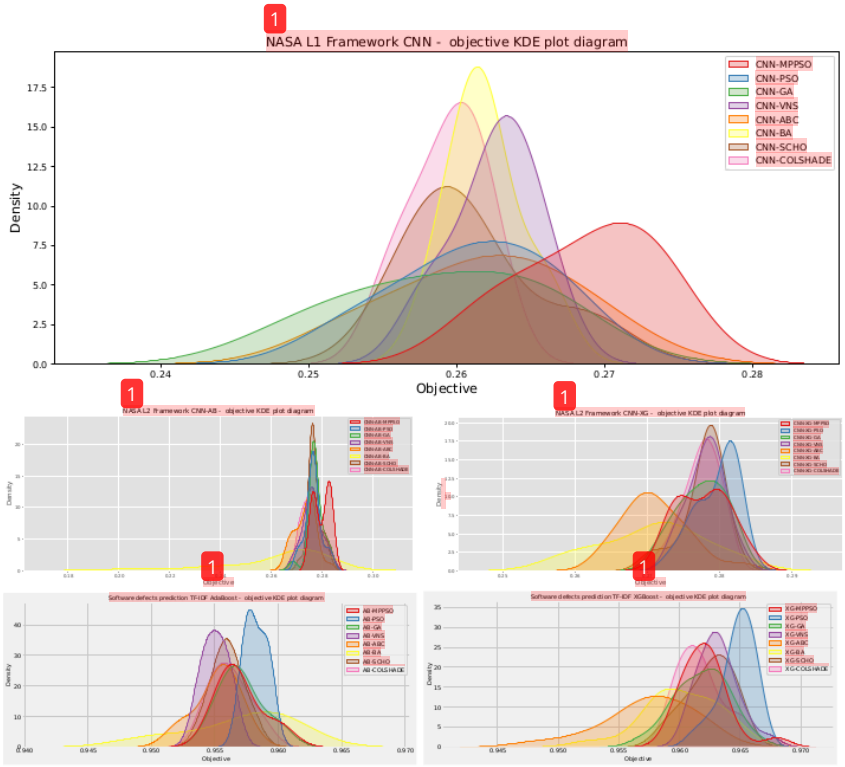
6.3.8 Статистичка валидација

Када се упоређени алгоритми ослањају на случајност, као код метахеуристика, од суштинског је значаја да се провери статистичка значајност резултата јер једна симулација можда није репрезентативна за укупне перформансе. Имајући ово у виду, симулације у овом раду су спроведене у 30 индивидуално независних серија и подаци су прикупљени за даљу анализу.

За валидацију статистичке значајности исхода могу се применити две врсте статистичких тестова, параметарски и непараметарски тест. Први корак укључује валидацију да ли је могуће одредити описано путем параметарских тестова или да ли треба применити непараметарску анализу. Употреба параметарских тестова је оправдана ако су испуњени услови независности, нормалности и хомоскедастичности [65]. Пошто се све симулације спроводе коришћењем независних насумичних семена, критеријуми независности су испуњени. Хомоскедастичност је потврђена Лавиновим тестом (енгл. - *Levene's test*) [74] који је резултирао са $p-values$ од 0.68 за све случајеве, што указује да је и овај услов испуњен.

Нормалност се процењује коришћењем Шапиро-Вилк теста [66], са $p-values$ израчунатим за сваки метод укључен у упоредну анализу. Пошто су све вредности испод прага од 0.05, нулта хипотеза X0 може бити одбачена, што сугерише да резултати не потичу из нормалних дистрибуција.

Ови налази су додатно поткрепљени *KDE* дијаграмима циљне функције представљеним Сликом 6.38.



Слика 6.38: *KDE* дијаграми пет експеримената предвиђања софтверске грешке.

Резултати Шапиро-Вилк теста су дати у Табели 6.39.

Табела 6.39: Резултати пет спроведених Шапиро-Вилко симулација за верификацију услова нормалности за безбедно коришћење параметарских тестова.

Model	*MPPSO*	*PSO*	*GA*	*VNS*	*ABC*	*BA*	*SCHO*	*COLSHADE*
CNN Optimization	0.033	0.029	0.023	0.017	0.035	0.044	0.041	0.037
AdaBoost Optimization	0.019	0.027	0.038	0.021	0.034	0.043	0.025	0.049
XGBoost Optimization	0.022	0.026	0.032	0.018	0.029	0.040	0.038	0.042
AdaBoost NLP Optimization	0.031	0.022	0.037	0.026	0.033	0.046	0.030	0.045
XGBoost NLP Optimization	0.028	0.024	0.034	0.019	0.031	0.041	0.036	0.043

Пошто један од критеријума потребних за употребу параметарских тестова није испуњен, даље евалуације се спроводе путем непараметарских тестова. Вилкоксонов тест упарених узорака се стога примењује за поређење перформанси *MPPSO* са другим алгоритмима укљученим у упоредну анализу.

Резултати ових тестова су приказани у Табели 6.40.

Табела 6.40: Резултати Вилкоксоновог теста упарених узорака за пет спроведених симулација.

MPPSO vs. others	*PSO*	*GA*	*VNS*	*ABC*	*BA*	*SCHO*	*COLSHADE*
CNN Optimization	0.027	0.024	0.034	0.018	0.032	0.040	0.045
AdaBoost Optimization	0.020	0.028	0.031	0.022	0.030	0.038	0.042
XGBoost Optimization	0.033	0.023	0.037	0.019	0.031	0.043	0.046
AdaBoost NLP Optimization	0.029	0.026	0.035	0.021	0.029	0.041	0.044
XGBoost NLP Optimization	0.024	0.027	0.032	0.020	0.034	0.039	0.047

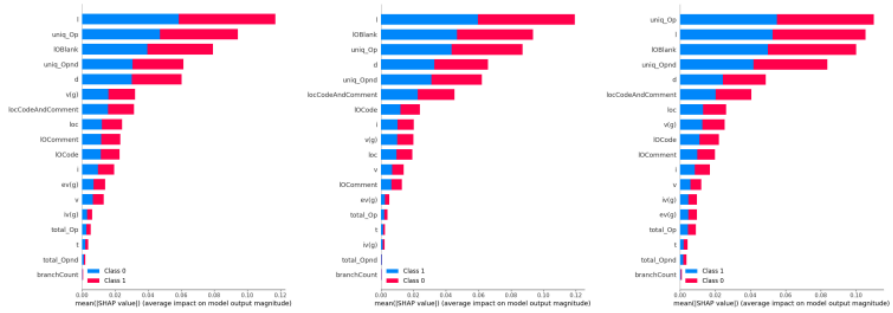
Праг од $\alpha = 0.05$ није прекорачен ни у једном од тест случајева што указује да резултати постигнути *MPPSO* алгоритмом имају статистички значајна побољшања у односу на друге евалуиране методе.

6.3.9 Интерпретације најбољег модела

Интерпретација модела игра све важнију улогу у савременим истраживањима у области вештачке интелигенције. Често разумевање фактора и њиховог степена утицаја на одлуку може помоћи да се боље разуме проблем који је у питању, као и да се открију било какве ненамерне пристрасности у подацима. Постоји неколико нових метода за тумачење модела, а за једноставније моделе ове интерпретације су прилично јасне. Међутим, компликованији модели и модели који користе вишеслојне оквире могу бити теже подложни тумачењу.

Овај рад користи *SHAP* анализу да би се утврдио значај и утицај карактеристика на оптимизоване одлуке модела. *SHAP* анализа користи приступ укорењен у теорији игара који третира карактеристике као такмичаре у кооперативном такмичењу. Допринос сваке карактеристике као и њихов заједнички допринос се узима у обзир приликом одређивања важности. Важност се такође може анализирати на глобалном и локалном нивоу чинећи *SHAP* свестраним и обећавајућим приступом за анализу модела.

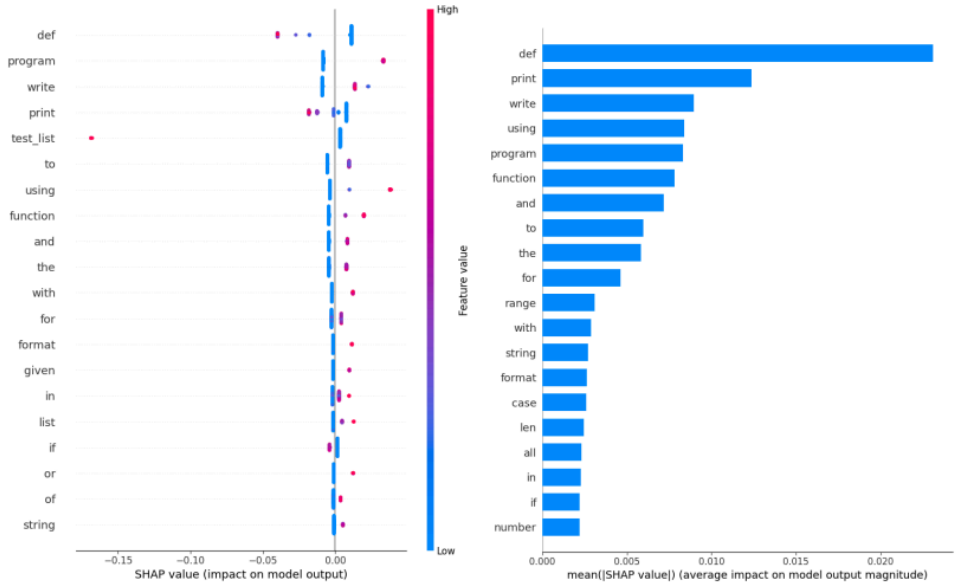
Значај карактеристика за моделе са најбољим перформансама конструисане у *L1* и *L2* симулацијама је дат Сликом 6.39.



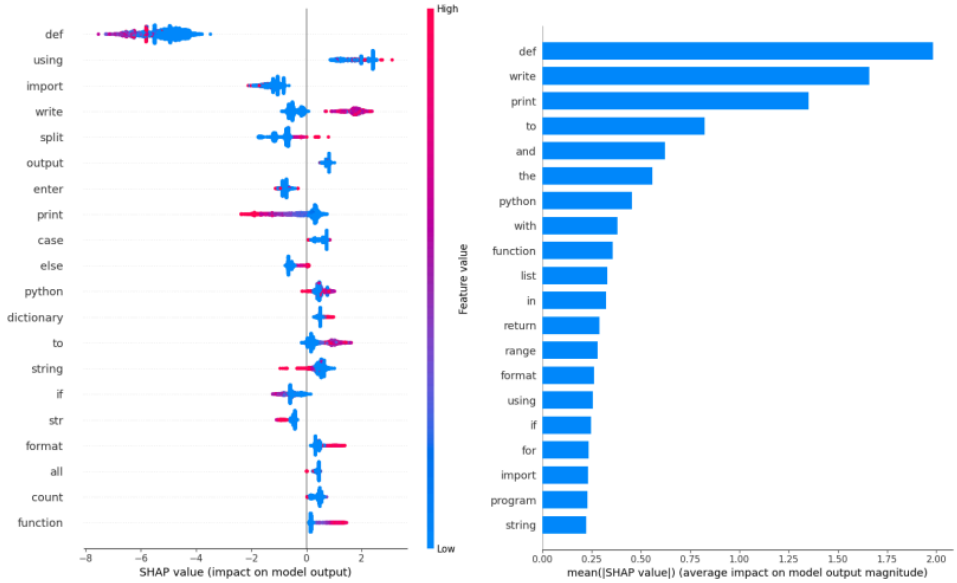
Слика 6.39: Важност обележја за најбоље моделе *CNN*, *XGBoost* и *AdaBoost* на HACA скупу података.

Сваки конструисани класификатор ставља мало другачији значај сваке карактеристике. Међутим, *I*, *Uniq_OP*, *IOBland* и *d* су универзално високо рангирани. Најбољи модели *CNN* и *AdaBoost* показују највећи утицај за *I* функцију, док је ова функција друга најбоља за *XGBoost* класификатор, са највећим значајем при датој особини *Uniq_OP*.

Значај карактеристика за моделе са најбољим перформансама у *NLP* симулацијама је такође приказан на Слици 6.40 за *AdaBoost* модел и на Слици 6.41 за *XGBoost* модел.



Слика 6.40: Важност обележја за најбоље моделе *AdaBoost* на *NLP* скупу података.



Слика 6.41: Важност обележја за најбоље моделе *XGBoost* на *NLP* скупу података.

TF-IDF векторизоване важности карактеристика се још једном тумаче коришћењем *SHAP* анализе и 20 најважнијих карактеристика приказаних за оба модела. Оба модела придају велики значај кључној речи *def*, као и кључним речима за штампање и писање где се грешке често могу појавити. *AdaBoost* модел придаје значајну важност *def*, при чему *print* држи друго место са знатно мањим значајем. Модел *XGBoost*, међутим, има равномернију дистрибуцију значаја. Док *def* има велики значај, други највећи утицај је на кључну реч *write* коју следи *print* са незнатно смањеном важности.

Разумевање важности карактеристика у одлукама о моделима може додатно помоћи у разумевању изазова повезаних са откривањем грешака у софтверу. Штавише, одређивање карактеристика које играју и важне улоге у овим класификацијама може додатно помоћи у смањењу рачунарских трошкова модела у примени, као и помоћи у побољшању прикупљања података у будућности. Откривање скривених пристрасности модела је такође од суштинског значаја за успостављање поверења у одлуке модела чиме се побољшава генерализација и објективност одлука.

Глава 7

Закључна разматрања

Проблем поморског саобраћаја поседује вишеструки значај за друштво. Унапређења у виду безбедности које би донео прецизнији и пре свега поузданији систем праћења пловила, што укључује и њихову идентификацију, не само да би учинила пловидбу која је захтевна по себи мање опасном већ ће и унапредити еколошке аспекте ове врсте саобраћаја. Проблеми безбедности се тичу непогодности путовања преко воде као што су временски услови и одсуство било каквих путоказа. Поред добро познатих проблема, пиратерија је такође један од проблема модерног поморског транспорта, на пример користећи деонице за које се зна да је сигнал за време пролаза кроз њих слабији, овакви индивидуалци би могли да искористе. Друге врсте злоупотреба се тичу намерног искључивања уређаја за праћење због вршења радњи које штете воденом свету. То су најчешће методе чијом употребом је могуће прикупити огромне количине морских плодова што доводи до озбиљних дисбаланса у природи.

Проблем предикције грешке у коду на први поглед можда изгледа мање важно у поређењу са проблемима који се тичу глобалног снабдевања путем поморских рута. Тако је само на први поглед јер је напретком технологије дошло до тога да ни бродови који преносе тај терет не могу да функционишу без поузданог кода. Систем за навигацију који користе пловила мора да ради без грешке и уски временски прозори у којима брод нема навигацију могу довести до трагичних последица. Поузданост програма, односно кода којим је програм написан ће само имати већу важност у будућности како се све више критичних и опасних послова ослања на њих.

Циљ истраживања је било унапређење *PSO* алгоритма за описане проблеме и резултат јесу две нове верзије овог алгоритма. Опширно тестирање је примењено за оба случаја као и статистичка валидација и анализа. Због *NFL* теореме није било могуће направити један алгоритам који решава оба ова проблема подједнако добро. Ову чињеницу подржава и то да је природа описана два проблема врло различита, што значи да су и подаци о њима јако различити. Приликом тестирања са осталим алгоритмима високих перформанси предложена решења постижу најбоље резултате, али пошто је опсег експеримената опширан као и категорија перформанси које су праћене, *NFL* теорема још једном долази до изражаја. Другим речима, одређене метахеуристике су показале пожељне афинитете који сугеришу потенцијал истраживања ових алгоритама. Проналажење бољих и новијих решења је нешто што је препознатљиво за област оптимизације те смер у којем је могуће наставити ово истраживање може бити путем истраживање већег броја алгоритама за решавање проблема описаних у истраживању. Такође је могуће експериментисати са додатним модификацијама *PSO* алгоритма како би још веће унапређење било постигнуто.

Библиографија

[1] M. Alenezi and M. Akour, "Ai-driven innovations in software engineering: A review of current practices and future directions," *Applied Sciences*, vol. 15, no. 3, p. 1344, 2025.

[2] D. Wolpert and W. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[3] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, vol. 4, pp. 1942–1948, ieee, 1995.

[4] P. Preux and E.-G. Talbi, "Towards hybrid evolutionary algorithms," *International transactions in operational research*, vol. 6, no. 6, pp. 557–570, 1999.

[5] T. Bezdan, A. Petrovic, M. Zivkovic, I. Strumberger, V. K. Devi, and N. Bacanin, "Current best opposition-based learning salp swarm algorithm for global numerical optimization," in *2021 zooming innovation in consumer technologies conference (ZINC)*, pp. 5–10, IEEE, 2021.

[6] M. Todorovic, A. Petrovic, A. Toskovic, M. Zivkovic, L. Jovanovic, and N. Bacanin, "Multivariate bitcoin price prediction based on lstm tuned by hybrid reptile search algorithm," in *2023 16th international conference on advanced Technologies, Systems and Services in Telecommunications (TELSIKS)*, pp. 195–198, IEEE, 2023.

[7] A. Petrovic, L. Jovanovic, N. Bacanin, M. Antonijevic, N. Savanovic, M. Zivkovic, M. Milovanovic, and V. Gajic, "Exploring metaheuristic optimized machine learning for software defect detection on natural language and classical datasets," *Mathematics*, vol. 12, no. 18, p. 2918, 2024.

[8] A. Petrovic, R. Damaševičius, L. Jovanovic, A. Toskovic, V. Simic, N. Bacanin, M. Zivkovic, and P. Spalevič, "Marine vessel classification and multivariate trajectories forecasting using metaheuristics-optimized extreme gradient boosting and recurrent neural networks," *Applied Sciences*, vol. 13, no. 16, p. 9181, 2023.

[9] N. Bacanin, R. Stoean, M. Zivkovic, A. Petrovic, T. A. Rashid, and T. Bezdan, "Performance of a novel chaotic firefly algorithm with enhanced exploration for tackling global optimization problems: Application for dropout regularization," *Mathematics*, vol. 9, no. 21, p. 2705, 2021.

[10] R. Damaševičius, L. Jovanovic, A. Petrovic, M. Zivkovic, N. Bacanin, D. Jovanovic, and M. Antonijevic, "Decomposition aided attention-based recurrent neural networks

for multistep ahead time-series forecasting of renewable power generation," *PeerJ Computer Science*, vol. 10, p. e1795, 2024.

[11] A. Minic, L. Jovanovic, N. Bacanin, C. Stoean, M. Zivkovic, P. Spalevic, A. Petrovic, M. Dobrojevic, and R. Stoean, "Applying recurrent neural networks for anomaly detection in electrocardiogram sensor data," *Sensors*, vol. 23, no. 24, p. 9878, 2023.

[12] N. Bacanin, V. Simic, M. Zivkovic, M. Alrasheedi, and A. Petrovic, "Cloud computing load prediction by decomposition reinforced attention long short-term memory network optimized by modified particle swarm optimization algorithm," *Annals of Operations Research*, pp. 1–34, 2023.

[13] M. Zivkovic, C. Stoean, A. Chhabra, N. Budimirovic, A. Petrovic, and N. Bacanin, "Novel improved salp swarm algorithm: An application for feature selection," *Sensors*, vol. 22, no. 5, p. 1711, 2022.

[14] N. Savanović, A. Toskovic, A. Petrovic, M. Zivkovic, R. Damaševičius, L. Jovanovic, N. Bacanin, and B. Nikolic, "Intrusion detection in healthcare 4.0 internet of things systems via metaheuristics optimized machine learning," *Sustainability*, vol. 15, no. 16, p. 12563, 2023.

[15] A. Petrovic, N. Bacanin, L. Jovanovic, J. Cadjenovic, J. Kaljevic, M. Zivkovic, and M. Antonijevic, "Computer-vision unmanned aerial vehicle detection system using yolov8 architectures," *International Journal of Robotics and Automation Technology*, vol. 11, pp. 1–12, 2024.

[16] A. Petrovic, L. Jovanovic, K. Venkatachalam, M. Zivkovic, N. Bacanin, and N. Budimirovic, "Anomaly detection in electrocardiogram signals using metaheuristic optimized time-series classification with attention incorporated models," *International Journal of Hybrid Intelligent Systems*, vol. 20, no. 2, pp. 159–183, 2024.

[17] X. Han, C. Armenakis, and M. Jadidi, "Modeling vessel behaviours by clustering ais data using optimized dbscan," *Sustainability*, vol. 13, no. 15, p. 8162, 2021.

[18] F. Huang, Jie andZhu, Z. Huang, J. Wan, and Y. Ren, "Research on real-time anomaly detection of fishing vessels in a marine edge computing environment," 2021.

[19] F. Yang, Y. Qiao, W. Wei, X. Wang, D. Wan, R. Damaševičius, and M. Woźniak, "Ddtree: A hybrid deep learning model for real-timewaterway depth prediction and smart navigation," *Applied Sciences*, vol. 10, no. 8, 2020.

[20] Y. Zhou, W. Daamen, T. Vellinga, and S. P. Hoogendoorn, "Ship classification based on ship behavior clustering from ais data," *Ocean Engineering*, vol. 175, pp. 176–187, 2019.

[21] X. Wang and Y. Xiao, "A deep learning model for ship trajectory prediction using automatic identification system (ais) data," *Information*, vol. 14, no. 4, 2023.

[22] S. Alyahyan, M. N. Alatawi, M. M. Alnfiai, S. D. Alotaibi, A. Alshammari, Z. Alzaid, and H. S. Alwageed, "Software reliability assessment: An architectural and component impact analysis," *Tsinghua Science and Technology*, 2024.

[23] H. Zhang, X.-Z. Gao, Z. Wang, and G. Wang, "Guest editorial of the special section on neural computing-driven artificial intelligence for consumer electronics," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 3517–3520, 2024.

[24] S. Mcmurray and A. H. Sodhro, "A study on ml-based software defect detection for security traceability in smart healthcare applications," *Sensors*, vol. 23, no. 7, p. 3470, 2023.

[25] G. Giray, K. E. Bennin, Ö. Köksal, Ö. Babur, and B. Tekinerdogan, "On the use of deep learning in software defect prediction," *Journal of Systems and Software*, vol. 195, p. 111537, 2023.

[26] J. R. Jim, M. A. R. Talukder, P. Malakar, M. M. Kabir, K. Nur, and M. Mridha, "Recent advancements and challenges of nlp-based sentiment analysis: A state-of-the-art review," *Natural Language Processing Journal*, p. 100059, 2024.

[27] C. Zhang, J. Chen, J. Li, Y. Peng, and Z. Mao, "Large language models for human-robot interaction: A review," *Biomimetic Intelligence and Robotics*, p. 100131, 2023.

[28] Y. Peng, M. He, F. Hu, Z. Mao, X. Huang, and J. Ding, "Predictive modeling of flexible ehd pumps using kolmogorov-arnold networks," *arXiv preprint arXiv:2405.07488*, 2024.

[29] V. Mizdrakovic, M. Kljajic, M. Zivkovic, N. Bacanin, L. Jovanovic, M. Deveci, and W. Pedrycz, "Forecasting bitcoin: Decomposition aided long short-term memory based time series modelling and its explanation with shapley values," *Knowledge-Based Systems*, p. 112026, 2024.

[30] M. Zivkovic, N. Bacanin, M. Antonijevic, B. Nikolic, G. Kvascev, M. Marjanovic, and N. Savanovic, "Hybrid cnn and xgboost model tuned by modified arithmetic optimization algorithm for covid-19 early diagnostics from x-ray images," *Electronics*, vol. 11, no. 22, p. 3798, 2022.

[31] M. Salb, L. Jovanovic, N. Bacanin, M. Antonijevic, M. Zivkovic, N. Budimirovic, and L. Abualigah, "Enhancing internet of things network security using hybrid cnn and xgboost model tuned via modified reptile search algorithm," *Applied Sciences*, vol. 13, no. 23, p. 12687, 2023.

[32] L. Jovanovic, D. Jovanovic, M. Antonijevic, B. Nikolic, N. Bacanin, M. Zivkovic, and I. Strumberger, "Improving phishing website detection using a hybrid two-level framework for feature selection and xgboost tuning," *Journal of Web Engineering*, vol. 22, no. 3, pp. 543–574, 2023.

[33] S. Mirjalili and S. Mirjalili, "Genetic algorithm," *Evolutionary algorithms and neural networks: theory and applications*, pp. 43–55, 2019.

[34] D. E. Goldberg, *Genetic algorithms*. pearson education India, 2013.

[35] D. Karaboga, "Artificial bee colony algorithm," *scholarpedia*, vol. 5, no. 3, p. 6915, 2010.

[36] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature inspired cooperative strategies for optimization (NICSO 2010)*, pp. 65–74, Springer, 2010.

[37] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in engineering software*, vol. 95, pp. 51–67, 2016.

[38] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Future generation computer systems*, vol. 97, pp. 849–872, 2019.

[39] M. V. Suntsova and A. A. Buzdin, "Differences between human and chimpanzee genomes and their implications in gene expression, protein functions and biochemical properties of the two species," *BMC genomics*, vol. 21, no. Suppl 7, p. 535, 2020.

[40] J. Bai, Y. Li, M. Zheng, S. Khatir, B. Benaissa, L. Abualigah, and M. A. Wahab, "A sinh cosh optimizer," *Knowledge-Based Systems*, vol. 282, p. 111081, 2023.

[41] L. A. H. S. Pinto, "Progress in artificial intelligence," 1978.

[42] L. Abualigah, A. H. Gandomi, M. A. Elaziz, A. G. Hussien, A. M. Khasawneh, M. Alshinwan, and E. H. Houssein, "Nature-inspired optimization algorithms for text document clustering—a comprehensive analysis," *Algorithms*, vol. 13, no. 12, p. 345, 2020.

[43] M. Zivkovic, A. Petrovic, N. Bacanin, M. Djuric, A. Vesic, I. Strumberger, and M. Marjanovic, "Training logistic regression model by hybridized multi-verse optimizer for spam email classification," in *Proceedings of International Conference on Data Science and Applications: ICDSA 2022, Volume 2*, pp. 507–520, Springer, 2023.

[44] B. Radomirovic, A. Petrovic, M. Zivkovic, A. Njegus, N. Budimirovic, and N. Bacanin, "Efficient spam email classification logistic regression model trained by modified social network search algorithm," in *Computational Intelligence and Blockchain in Complex Systems*, pp. 39–55, Elsevier, 2024.

[45] A. Hosseinalipour and R. Ghanbarzadeh, "A novel metaheuristic optimisation approach for text sentiment analysis," *International journal of machine learning and cybernetics*, vol. 14, no. 3, pp. 889–909, 2023.

[46] S. Shah, H. Ghomeshi, E. Vakaj, E. Cooper, and S. Fouad, "A review of natural language processing in contact centre automation," *Pattern Analysis and Applications*, vol. 26, no. 3, pp. 823–846, 2023.

[47] M. N. Ashtiani and B. Raahemi, "News-based intelligent prediction of financial markets using text mining and machine learning: A systematic literature review," *Expert Systems with Applications*, vol. 217, p. 119509, 2023.

[48] A. Shah, K. Kothari, U. Thakkar, and S. Khara, "User review classification and star rating prediction by sentimental analysis and machine learning classifiers," in *Information and Communication Technology for Sustainable Development: Proceedings of ICT4SD 2018*, pp. 279–288, Springer, 2020.

[49] J. Devlin, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[50] S. Altan and H. Shah, "A survey on bert and its applications," in *2023 20th Learning and Technology Conference (L&T)*, pp. 161–166, IEEE, 2023.

[51] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[52] Y. LeCun, Y. Bengio, *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[53] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proceedings of the 30th International Conference on Machine Learning* (S. Dasgupta and D. McAllester, eds.), vol. 28 of *Proceedings of Machine Learning Research*, (Atlanta, Georgia, USA), pp. 1310–1318, PMLR, 17–19 Jun 2013.

[54] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *European conference on computational learning theory*, pp. 23–37, Springer, 1995.

[55] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

[56] A. W. Mohamed, A. A. Hadi, A. K. Mohamed, and N. H. Awad, "Evaluating the performance of adaptive gainingsharing knowledge based algorithm on cec 2020 benchmark problems," in *2020 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, IEEE, 2020.

[57] K. Yu, J. Liang, B. Qu, X. Chen, and H. Wang, "Parameters identification of photovoltaic models using an improved jaya optimization algorithm," *Energy Conversion and Management*, vol. 150, pp. 742–753, 2017.

[58] A. V. Chechkin, R. Metzler, J. Klafter, and V. Y. Gonchar, "Introduction to the theory of lévy flights," *Anomalous transport: Foundations and applications*, pp. 129–162, 2008.

[59] X.-S. Yang and S. Deb, "Cuckoo search via lévy flights," in *2009 World congress on nature & biologically inspired computing (NaBIC)*, pp. 210–214, Ieee, 2009.

[60] X.-S. Yang, *Nature-inspired metaheuristic algorithms*. Luniver press, 2010.

[61] X.-S. Yang, "Optimization and metaheuristic algorithms in engineering," *Metaheuristics in water, geotechnical and transport engineering*, vol. 1, p. 23, 2013.

[62] M. Abdel-Basset, L. Abdel-Fatah, and A. K. Sangaiah, "Metaheuristic algorithms: A comprehensive review," *Computational intelligence for multimedia big data on the cloud with engineering applications*, pp. 185–231, 2018.

[63] A. Kazikova, M. Pluhacek, and R. Senkerik, "How does the number of objective function evaluations impact our understanding of metaheuristics behavior?," *IEEE Access*, vol. 9, pp. 44032–44048, 2021.

[64] M. Khishe and M. R. Mosavi, "Chimp optimization algorithm," *Expert systems with applications*, vol. 149, p. 113338, 2020.

[65] A. LaTorre, D. Molina, E. Osaba, J. Poyatos, J. Del Ser, and F. Herrera, "A prescription of methodological guidelines for comparing bio-inspired optimization algorithms," *Swarm and Evolutionary Computation*, vol. 67, p. 100973, 2021.

[66] S. S. Shapiro and R. Francia, "An approximate analysis of variance test for normality," *Journal of the American statistical Association*, vol. 67, no. 337, pp. 215–216, 1972.

[67] F. Wilcoxon, "Individual comparisons by ranking methods," in *Breakthroughs in statistics*, pp. 196–202, Springer, 1992.

[68] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 4768–4777, Curran Associates Inc., 2017.

[69] Y. Yuan, L. Wu, and X. Zhang, "Gini-impurity index analysis," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3154–3169, 2021.

[70] T. McCabe, "A complexity measure," *IEEE Transactions on Software Engineering*, vol. 2, pp. 308–320, December 1976.

[71] M. Halstead, *Elements of Software Science*. Elsevier, 1977.

[72] P. Hansen, N. Mladenović, J. Brimberg, and J. A. M. Pérez, "Variable neighborhood search," in *Handbook of metaheuristics*, pp. 57–97, Springer, 2018.

[73] J. Gurrola-Ramos, A. Hernàndez-Aguirre, and O. Dalmau-Cedeño, "Colshade for real-world single-objective constrained optimization problems," in *2020 IEEE congress on evolutionary computation (CEC)*, pp. 1–8, IEEE, 2020.

[74] G. V. Glass, "Testing homogeneity of variances," *American Educational Research Journal*, vol. 3, no. 3, pp. 187–190, 1966.

УНАПРЕЂЕЊЕ АЛГОРИТАМА РОЈЕВА ЧЕСТИЦА ЗА ПРОБЛЕМЕ КЛАСИФИКАЦИЈЕ И РЕГРЕСИЈЕ СА ПРИМЕРИМА ПРИМЕНЕ

ORIGINALITY REPORT

21%
SIMILARITY INDEX

PRIMARY SOURCES

1	Aleksandar Petrovic, Luka Jovanovic, Nebojsa Bacanin, Milos Antonijevic et al. "Exploring Metaheuristic Optimized Machine Learning for Software Defect Detection on Natural Language and Classical Datasets", Mathematics, 2024 Crossref	2677 words — 7%
2	Aleksandar Petrovic, Robertas Damaševičius, Luka Jovanovic, Ana Toskovic et al. "Marine Vessel Classification and Multivariate Trajectories Forecasting Using Metaheuristics-Optimized eXtreme Gradient Boosting and Recurrent Neural Networks", Applied Sciences, 2023 Crossref	1932 words — 5%
3	mdpi-res.com Internet	422 words — 1%
4	vigir.ee.missouri.edu Internet	165 words — < 1%
5	downloads.hindawi.com Internet	135 words — < 1%
6	arxiv.org Internet	105 words — < 1%
7	eng.singidunum.ac.rs Internet	105 words — < 1%

8	mcf.gsfc.nasa.gov Internet	100 words — < 1%
9	peerj.com Internet	85 words — < 1%
10	www.mdpi.com Internet	84 words — < 1%
11	go.gale.com Internet	81 words — < 1%
12	"Data Intelligence and Cognitive Informatics", Springer Science and Business Media LLC, 2023 Crossref	80 words — < 1%
13	assets-eu.researchsquare.com Internet	59 words — < 1%
14	Nebojsa Bacanin, Mirjana Perisic, Gordana Jovanovic, Robertas Damaševičius et al. "The explainable potential of coupling hybridized metaheuristics, XGBoost, and SHAP in revealing toluene behavior in the atmosphere", Science of The Total Environment, 2024 Crossref	57 words — < 1%
15	e-space.mmu.ac.uk Internet	50 words — < 1%
16	uvidok.rcub.bg.ac.rs Internet	50 words — < 1%
17	www.researchgate.net Internet	48 words — < 1%
18	nardus.mpn.gov.rs Internet	46 words — < 1%

19	Vule Mizdrakovic, Maja Kljajic, Miodrag Zivkovic, Nebojsa Bacanin, Luka Jovanovic, Muhammet Deveci, Witold Pedrycz. "Forecasting bitcoin: Decomposition aided long short-term memory based time series modeling and its explanation with Shapley values", Knowledge-Based Systems, 2024 Crossref	44 words — < 1%
20	Jelica Cincovic, Luka Jovanovic, Bosko Nikolic, Nebojsa Bacanin. "Neurodegenerative Condition Detection Using Modified Metaheuristic for Attention Based Recurrent Neural Networks and Extreme Gradient Boosting Tuning", IEEE Access, 2024 Crossref	41 words — < 1%
21	www.taci-journal.org Internet	40 words — < 1%
22	www.insightsociety.org Internet	38 words — < 1%
23	Nebojsa Bacanin, Luka Jovanovic, Miodrag Zivkovic, Mohamed Salb, Ali Elsadai, Marko Sarac. "Decomposition aided cloud load forecasting with optimized long-short term memory networks", 2023 16th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS), 2023 Crossref	35 words — < 1%
24	Vishal Saini, Sunil Kumar Singal, Rajeshwer Prasad Saini. "Renewable energy-based system for electrification of remote villages", 2022 International Conference on Green Energy, Computing and Sustainable Technology (GECOST), 2022 Crossref	35 words — < 1%
25	library.acadlore.com Internet	35 words — < 1%

26	pubs2.ascee.org Internet	34 words — < 1%
27	www.nature.com Internet	32 words — < 1%
28	ebin.pub Internet	30 words — < 1%
29	pdfs.semanticscholar.org Internet	30 words — < 1%
30	P. Umaeswari, R. Sonia, T.R. Saravanan, N. Poyyamozhi. "IoT-Enabled energy conservation in residential Buildings: Machine learning models for analyzing annual solar power consumption", Solar Energy, 2024 Crossref	29 words — < 1%
31	www.db-thueringen.de Internet	29 words — < 1%
32	www.medrxiv.org Internet	27 words — < 1%
33	Safet Purkovic, Luka Jovanovic, Miodrag Zivkovic, Milos Antonijevic et al. "Audio analysis with convolutional neural networks and boosting algorithms tuned by metaheuristics for respiratory condition classification", Journal of King Saud University - Computer and Information Sciences, 2024 Crossref	25 words — < 1%
34	Nemanja Milutinovic, Masa Gajevic, Jelena Krstovic, Aleksandar Petrovic, Nebojsa Bacanin, Milos Antonijevic. "Performance of Arithmetic Optimization Algorithm for ELM Tuning Applied to IoT security", 2022 30th Telecommunications Forum (TELFOR), 2022 Crossref	24 words — < 1%

35	fedorabg.bg.ac.rs Internet	24 words — < 1%
36	www.bpasjournals.com Internet	24 words — < 1%
37	dspacemainprd01.lib.uwaterloo.ca Internet	23 words — < 1%
38	norlx65.nordita.org Internet	23 words — < 1%
39	John Philipose Villoth, Miodrag Zivkovic, Tamara Zivkovic, Mahmoud Abdel-salam et al. "Two-tier deep and machine learning approach optimized by adaptive multi-population firefly algorithm for software defects prediction", Neurocomputing, 2025 Crossref	22 words — < 1%
40	Luka Jovanovic, Nemanja Milutinovic, Masa Gajevic, Jelena Krstovic, Tarik A. Rashid, Aleksandar Petrovic. "Sine Cosine Algorithm for Simple recurrent neural network Tuning for Stock Market Prediction", 2022 30th Telecommunications Forum (TELFOR), 2022 Crossref	22 words — < 1%
41	Marco A.R.S. Cruz, Helder R.O. Rocha, Marcia H.M. Paiva, Marcelo E.V. Segatto, Eglantine Camby, Gilles Caporossi. "An algorithm for cost optimization of PMU and communication infrastructure in WAMS", International Journal of Electrical Power & Energy Systems, 2019 Crossref	22 words — < 1%
42	Rehab Ali Ibrahim, Mohamed Aly Saad Aly, Yasmine S. Moemen, Ibrahim El Tantawy El Sayed et al. "Boosting Sinh Cosh Optimizer and Arithmetic	22 words — < 1%

Optimization Algorithm for Improved Prediction of Biological Activities for Indoloquinoline Derivatives", Chemosphere, 2024
Crossref

43	dk.um.si Internet	22 words — < 1%
44	msesj.com Internet	22 words — < 1%
45	Melissa N. Melaugh, Sonya Coleman, Dermot Kerr. "A Computational Approach to Uncertainty in DNA Sequences", 2023 IEEE Symposium Series on Computational Intelligence (SSCI), 2023 Crossref	21 words — < 1%
46	Nikola Savanović, Ana Toskovic, Aleksandar Petrovic, Miodrag Zivkovic et al. "Intrusion Detection in Healthcare 4.0 Internet of Things Systems via Metaheuristics Optimized Machine Learning", Sustainability, 2023 Crossref	21 words — < 1%
47	doaj.org Internet	21 words — < 1%
48	eteze.bg.ac.rs Internet	21 words — < 1%
49	publications.polymtl.ca Internet	21 words — < 1%
50	www.ijsr.net Internet	21 words — < 1%
51	Lecture Notes in Computer Science, 2007. Crossref	19 words — < 1%
52	web.archive.org Internet	

19 words — < 1%

53 www.informatica.si
Internet
19 words — < 1%

54 www.propulsiontechjournal.com
Internet
19 words — < 1%

55 deepai.org
Internet
18 words — < 1%

56 etd.repository.ugm.ac.id
Internet
18 words — < 1%

57 hdl.handle.net
Internet
18 words — < 1%

58 www.readkong.com
Internet
18 words — < 1%

59 Jiafen Lan, Mao Zheng, Zhiyuan Wang, Xiumin Chu, Hao Wu. "Intelligent Navigation Decisions-Making in Curved Channels: Review and Challenges", 2023 7th International Conference on Transportation Information and Safety (ICTIS), 2023
Crossref
17 words — < 1%

60 Miodrag Zivkovic, Luka Jovanovic, Marijana Pavlov, Nebojsa Bacanin, Milos Dobrojevic, Mohamed Salb. "Optimized recurrent neural networks with attention for wind farm energy generation forecasting", 2023 16th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS), 2023
Crossref
17 words — < 1%

61 Miodrag Zivkovic, Nebojsa Bacanin, Milos Antonijevic, Bosko Nikolic, Goran Kvascev, Marina Marjanovic, Nikola Savanovic. "Hybrid CNN and XGBoost Model
17 words — < 1%

Tuned by Modified Arithmetic Optimization Algorithm for COVID-19 Early Diagnostics from X-ray Images", Electronics, 2022
Crossref

62 Mohammed H. Qais, Hany M. Hasanien, Saad Alghuwainem. "Enhanced whale optimization algorithm for maximum power point tracking of variable-speed wind generators", Applied Soft Computing, 2020
Crossref
17 words — < 1%

63 Rehab Ali Ibrahim, Dalia Yousri, Mohamed Abd Elaziz, Samah Alshathri, Ibrahim Attiya. "Fractional Calculus-Based Slime Mould Algorithm for Feature Selection Using Rough Set", IEEE Access, 2021
Crossref
17 words — < 1%

64 d-nb.info
Internet
17 words — < 1%

65 ore.exeter.ac.uk
Internet
17 words — < 1%

66 tede.ufam.edu.br
Internet
17 words — < 1%

67 theses.lib.polyu.edu.hk
Internet
17 words — < 1%

68 ouci.dntb.gov.ua
Internet
16 words — < 1%

69 thesai.org
Internet
16 words — < 1%

70 www.ftn.uns.ac.rs
Internet
16 words — < 1%

71	Bessieres, D.. "High pressure thermal expansion of gases: measurements and calibration", Thermochimica Acta, 200504 Crossref	15 words — < 1%
72	Catalin Stoean, Miodrag Zivkovic, Aleksandra Bozovic, Nebojsa Bacanin et al. "Metaheuristic-Based Hyperparameter Tuning for Recurrent Deep Learning: Application to the Prediction of Solar Energy Generation", Axioms, 2023 Crossref	15 words — < 1%
73	Wang, Bo, Xiaohua Xia, and Jiangfeng Zhang. "A multi-objective optimization model for the life-cycle cost analysis and retrofitting planning of buildings", Energy and Buildings, 2014. Crossref	15 words — < 1%
74	hrcak.srce.hr Internet	15 words — < 1%
75	oa.upm.es Internet	15 words — < 1%
76	tel.archives-ouvertes.fr Internet	15 words — < 1%
77	www.jcobayliss.net Internet	15 words — < 1%
78	"Innovations in Cybersecurity and Data Science", Springer Science and Business Media LLC, 2024 Crossref	14 words — < 1%
79	Ana Toskovic, Aleksandar Petrovic, Luka Jovanovic, Nebojsa Bacanin, Miodrag Zivkovic, Milos Dobrojevic. "Chapter 5 Marine Vessel Trajectory Forecasting Using Long Short-Term Memory Neural Networks	14 words — < 1%

Optimized via Modified Metaheuristic Algorithm", Springer Science and Business Media LLC, 2024
Crossref

80 Xin Gao, Tianjun Li, Pramod Shukla. "Cosmological observables in multi-field inflation with a non-flat field space", Journal of Cosmology and Astroparticle Physics, 2014
Crossref
14 words — < 1%

81 dementia.talkbank.org
Internet
14 words — < 1%

82 link.springer.com
Internet
14 words — < 1%

83 singipedia.singidunum.ac.rs
Internet
14 words — < 1%

84 "Intelligent Systems and Pattern Recognition", Springer Science and Business Media LLC, 2025
Crossref
13 words — < 1%

85 "Intelligent and Fuzzy Systems", Springer Science and Business Media LLC, 2022
Crossref
13 words — < 1%

86 Oveis Abedinia, Ali Ghasemi-Marzbali, Venera Nurmanova, Mehdi Bagheri. "A New Reconfigured Electricity Market Bidding Strategy in View of Players' Concerns", IEEE Transactions on Industry Applications, 2022
Crossref
13 words — < 1%

87 paper.ijcsns.org
Internet
13 words — < 1%

88 portalcris.vdu.lt
Internet
13 words — < 1%

89	www.biorxiv.org Internet	13 words — < 1%
90	www.diva-portal.org Internet	13 words — < 1%
91	yokohamapublishers.jp Internet	13 words — < 1%
92	Loukas Katikas, Sofia Reizopoulou, Paraskevi Drakopoulou, Vassiliki Vassilopoulou. "Using fishery-related data, scientific expertise, and machine learning to improve marine habitat mapping in northeastern Mediterranean waters", Ecological Informatics, 2025 Crossref	12 words — < 1%
93	api.deepai.org Internet	12 words — < 1%
94	dokumen.pub Internet	12 words — < 1%
95	fr.scribd.com Internet	12 words — < 1%
96	publikace.k.utb.cz Internet	12 words — < 1%
97	repository.ufrpe.br Internet	12 words — < 1%
98	upcommons.upc.edu Internet	12 words — < 1%
99	docplayer.hu Internet	11 words — < 1%
100	journals.riverpublishers.com Internet	11 words — < 1%

101	www.cell.com Internet	11 words — < 1%
102	www.hindawi.com Internet	11 words — < 1%
103	www.tdx.cat Internet	11 words — < 1%
104	Djordje Mladenovic, Milos Antonijevic, Luka Jovanovic, Vladimir Simic et al. "Sentiment classification for insider threat identification using metaheuristic optimized machine learning classifiers", Scientific Reports, 2024 Crossref	10 words — < 1%
105	Fei Yang, Yanchen Wang, Yudong Guo, Haihang Jiang, Zhenxing Yao. "Reliability Evaluation and Its Influence on Traffic Application", Springer Science and Business Media LLC, 2025 Crossref	10 words — < 1%
106	eprints.gla.ac.uk Internet	10 words — < 1%
107	essay.utwente.nl Internet	10 words — < 1%
108	services.phaidra.univie.ac.at Internet	10 words — < 1%
109	www.ccas.ru Internet	10 words — < 1%
110	Nenad Stekić. "Military interventions in the unipolarity: Lessons for security dynamics of US-China relations amid Trump's second term", Politika nacionalne bezbednosti, 2025	9 words — < 1%

Crossref

111	dspace.lib.uom.gr Internet	9 words — < 1%
112	ijeecs.iaescore.com Internet	9 words — < 1%
113	raf.edu.rs Internet	9 words — < 1%
114	sd-www.jhuapl.edu Internet	9 words — < 1%
115	www.ybook.co.jp Internet	9 words — < 1%
116	Dusan Nikolic. "Climate changes and civil law: Elements for an adaptation strategy", Zbornik radova Pravnog fakulteta, Novi Sad, 2013 Crossref	8 words — < 1%
117	Fábio Luiz Usberti, Paulo Morelato França, André Luiz Morelato França. "GRASP with evolutionary path-relinking for the capacitated arc routing problem", Computers & Operations Research, 2013 Crossref	8 words — < 1%
118	Hao Liu, Hongbin Dong, Jing Zhou. "Feature Selection Method Based on Hybrid Multi-Strategy Pelican Optimization Algorithm", 2023 6th International Conference on Data Storage and Data Engineering (DSDE), 2023 Crossref	8 words — < 1%
119	annals-csis.org Internet	8 words — < 1%
120	dergipark.org.tr Internet	8 words — < 1%

121	mafiadoc.com Internet	8 words — < 1%
122	tama.green.gifu-u.ac.jp Internet	8 words — < 1%
123	tud.qucosa.de Internet	8 words — < 1%
124	vdoc.pub Internet	8 words — < 1%
125	www.coursehero.com Internet	8 words — < 1%
126	www.grafiati.com Internet	8 words — < 1%
127	Noorbakhsh Amiri Golilarz, Hui Gao, Hasan Demirel. "Satellite Image De-Noising With Harris Hawks Meta Heuristic Optimization Algorithm and Improved Adaptive Generalized Gaussian Distribution Threshold Function", IEEE Access, 2019 Crossref	7 words — < 1%
128	Ana Čulić, Sandro Nižetić, Jelena Čulić Gambiroža, Petar Šolić. "Progress in Data-Driven Thermal Comfort Analysis and Modeling", Energy and Buildings, 2025 Crossref	6 words — < 1%
129	Marko Stankovic, Luka Jovanovic, Aleksandra Bozovic, Nebojsa Budimirovic, Miodrag Zivkovic, Nebojsa Bacanin. "Exploring the potential of combining Mel spectrograms with neural networks optimized by the modified crayfish optimization algorithm for acoustic speed violation identification", International Journal of Hybrid Intelligent Systems, 2024 Crossref	6 words — < 1%

130	Peter .J. Bickel, Kjell A. Doksum. "Mathematical Statistics - Basic Ideas and Selected Topics, Volumes I-II Package", Chapman and Hall/CRC, 2019 Publications	6 words — < 1%
131	Timea Bezdan, Aleksandar Petrovic, Miodrag Zivkovic, Ivana Strumberger, V Kanchana Devi, Nebojsa Bacanin. "Current Best Opposition-Based Learning Salp Swarm Algorithm for Global Numerical Optimization", 2021 Zooming Innovation in Consumer Technologies Conference (ZINC), 2021 Crossref	6 words — < 1%
132	Mwewa Chikonkolo Mwape, Aditya Parmar, Franz Roman, Naushad M. Emmambux, Yaovi Ouézou Azouma, Oliver Hensel. "Life cycle sustainability assessment of staple food processing: A double and dynamic materiality approach", Sustainable Production and Consumption, 2025 Crossref	5 words — < 1%
133	journals.plos.org Internet	5 words — < 1%
134	researchportal.hkr.se Internet	5 words — < 1%
135	zealpress.com Internet	4 words — < 1%

EXCLUDE QUOTES	ON	EXCLUDE SOURCES	OFF
EXCLUDE BIBLIOGRAPHY	ON	EXCLUDE MATCHES	OFF