УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У НОВОМ САДУ

Анђела Жунић

Систем за детекцију аномалија на ивици интерната базиран на окружењу MPT-FLA

МАСТЕР РАД

Нови Сад, 2026.

УНИВЕРЗИТЕТ У НОВОМ САДУ ● **ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА**
21000 НОВИ САД, Трг Доситеја Обрадовића 6

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, **РБР**:			
Идентификациони број, **ИБР**:			
Тип документације, **ТД**:		Монографска документација	
Тип записа, **ТЗ**:		Текстуални штампани материјал	
Врста рада, **ВР**:		Дипломски – мастер рад	
Аутор, **АУ**:		Анђела Жунић	
Ментор, **МН**:		проф. др Мирослав Поповић	
Наслов рада, **НР**:		Систем за детекцију аномалија на ивици интернета базиран на окружењу MPT-FLA	
Језик публикације, **ЈП**:		Српски / латиница	
Језик извода, **ЈИ**:		Српски	
Земља публиковања, **ЗП**:		Република Србија	
Уже географско подручје, **УГП**:		Војводина	
Година, **ГО**:		2026	
Издавач, **ИЗ**:		Ауторски репринт	
Место и адреса, **МА**:		Нови Сад; трг Доситеја Обрадовића 6	
Физички опис рада, **ФО**: (поглавља/страна/ цитата/табела/слика/графика/прилога)		8/41/15/5/15/0/0	
Научна област, **НО**:		Електротехника и рачунарство	
Научна дисциплина, **НД**:		Рачунарска техника	
Предметна одредница/Кључне речи, **ПО**:		Системи на ивици интернета, Федеративно учење, Изолационе шуме, Детекција аномалија, MPT-FLA	
УДК			
Чува се, **ЧУ**:		У библиотеци Факултета техничких наука, Нови Сад	
Важна напомена, **ВН**:			
Извод, **ИЗ**:		У овом раду приказана је адаптација алгоритма за детекцију аномалија, заснованог на изолационим шумама, са PTB-FLA окружења на MPT-FLA окружење намењено извршавању на MicroPython на Raspberry Pi Pico W уређајима. Прилагођавање алгоритма обухватало је решавање ограничења меморије и процесорских ресурса карактеристичних за уграђене системе, уз очување функционалности оригиналне имплементације. Поред тога, реализован је систем са наизменичним режимима обучавања и закључивања, који омогућава детекцију аномалија над подацима прикупљеним са температурног сензора. Предложено решење експериментално је евалуирано на једном рачунару, на локалној рачунарској мрежи и на Raspberry Pi Pico W уређајима, пратећи уобичајене фазе у инжењерингу система заснованих на рачунару.	
Датум прихватања теме, **ДП**:			
Датум одбране, **ДО**:			
Чланови комисије, **КО**:	Председник:	проф. др Силвиа Гилезан	
	Члан:	проф. др Илија Башичевић	Потпис ментора
	Члан, ментор:	проф. др Мирослав Поповић	

UNIVERSITY OF NOVI SAD ● **FACULTY OF TECHNICAL SCIENCES**
21000 NOVI SAD, Trg Dositeja Obradovića 6

KEY WORDS DOCUMENTATION

Accession number, **ANO**:			
Identification number, **INO**:			
Document type, **DT**:		Monographic publication	
Type of record, **TR**:		Textual printed material	
Contents code, **CC**:		Master Thesis	
Author, **AU**:		Anđela Žunić	
Mentor, **MN**:		Miroslav Popović, PhD	
Title, **TI**:		Anomaly detection system at the edge of the internet based on the MPT-FLA	
Language of text, **LT**:		Serbian	
Language of abstract, **LA**:		Serbian	
Country of publication, **CP**:		Republic of Serbia	
Locality of publication, **LP**:		Vojvodina	
Publication year, **PY**:		2026	
Publisher, **PB**:		Author's reprint	
Publication place, **PP**:		Novi Sad, Dositeja Obradovica sq. 6	
Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes)		8/41/15/5/15/0/0	
Scientific field, **SF**:		Electrical Engineering	
Scientific discipline, **SD**:		Computer Engineering, Engineering of Computer Based Systems	
Subject/Key words, **S**/**KW**:		Edge systems, Federated Learning, Isolation Forests, Anomaly Detection, MPT-FLA	
UC			
Holding data, **HD**:		The Library of Faculty of Technical Sciences, Novi Sad, Serbia	
Note, **N**:			
Abstract, **AB**:		This thesis presents the adaptation of an anomaly detection algorithm based on Isolation Forest from the PTB-FLA framework to the MPT-FLA framework, enabling its execution on MicroPython on Raspberry Pi Pico W devices. The adaptation addressed the memory and computational constraints of embedded systems while preserving the functionality of the original implementation. In addition, a system with alternating training and inference modes was developed, enabling anomaly detection on data collected from the onboard temperature sensor. The proposed solution was experimentally evaluated on a single host, across a local network of hosts, and on Raspberry Pi Pico W devices by following the common phases of the engineering of computer based systems.	
Accepted by the Scientific Board on, **ASB**:			
Defended on, **DE**:			
Defended Board, **DB**:	President:	Silvia Ghilezan, PhD	
	Member:	Ilija Bašičević, PhD	Menthor's sign
	Member, Mentor:	Miroslav Popović, PhD	

	УНИВЕРЗИТЕТ У НОВОМ САДУ ● **ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА** 21000 НОВИ САД, Трг Доситеја Обрадовића 6	Број:
	ЗАДАТАК ЗА ЗАВРШНИ РАД	Датум:

Студијски програм:	Рачунарство и аутоматика		
Студент:	Анђела Жунић	Број индекса:	E2 72/2023
Степен и врста студија:	Мастер академске студије		
Област:	Електротехника и рачунарство		
Ментор:	проф. др Мирослав Поповић		

НА ОСНОВУ ПОДНЕТЕ ПРИЈАВЕ, ПРИЛОЖЕНЕ ДОКУМЕНТАЦИЈЕ И ОДРЕДБИ СТАТУТА ФАКУЛТЕТА ИЗДАЈЕ СЕ ЗАДАТАК ЗА ЗАВРШНИ РАД, СА СЛЕДЕЋИМ ЕЛЕМЕНТИМА:

- проблем – тема рада;
- начин решавања проблема и начин практичне провере резултата рада, ако је таква провера неопходна;

НАСЛОВ ЗАВРШНОГ РАДА:

Систем за детекцију аномалија на ивици интернета базиран на окружењу MPT-FLA

ТЕКСТ ЗАДАТКА:

У оквиру овог мастер рада потребно је урадити следеће:

1. Истражити теоријске основе федеративног учења и постојећи алгоритам за детекцију аномалија (ADA).
2. Дати преглед PTB-FLA и MPT-FLA окружења за развој алгоритама федеративног учења.
3. Превести постојећи ADA са PTB-FLA окружења на MPT-FLA окружење.
4. Евалуирати ADA базиран на MPT-FLA окружењу (ADA-MPT-FLA) на једном рачунару.
5. Евалуирати ADA-MPT-FLA на локалној рачунарској мрежи (LAN).
6. Прилагодити ADA-MPT-FLA за RPi Pico W уређаје (ADA-MPT-FLA-RPi).
7. Евалуирати ADA-MPT-FLA-RPi на ивичној мрежи заснованој на RPi Pico W уређајима.
8. Имплементирати систем са наизменичним режимом обучавања и закључивања у оквиру ADA-MPT-FLA-RPi.
9. Анализирати добијене резултате и извести закључке о перформансама и променљивости предложеног приступа у реалним системима на ивици интернета.

Руководилац студијског програма:	Ментор рада:

Примерак за: ◯ - Студента; ◯ - Ментора

УНИВЕРЗИТЕТ У НОВОМ САДУ ● ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
21000 НОВИ САД, Трг Доситеја Обрадовић 6

ИЗЈАВА О НЕПОСТОЈАЊУ СУКОБА ИНТЕРЕСА

Изјављујем да нисам у сукобу интереса у односу ментор – кандидат и да нисам члан породице (супружник или ванбрачни партнер, родитељ или усвојитељ, дете или усвојеник), повезано лице (крвни сродник ментора/кандидата у правој линији, односно у побочној линији закључно са другим степеном сродства, као ни физичко лице које се према другим основама и околностима може оправдано сматрати интересно повезаним са ментором или кандидатом), односно да нисам зависан/на од ментора/кандидата, да не постоје околности које би могле да утичу на моју непристрасност, нити да стичем било какве користи или погодности за себе или друго лице било позитивним или негативним исходом, као и да немам приватни интерес који утиче, може да утиче или изгледа као да утиче на однос ментор-кандидат.

У Новом Саду, дана _____

Ментор

_____

Кандидат

_____

Захвалност

Захваљујем се професору др. Мирославу Поповићу на стручној помоћи приликом израде мастер рада, као и на целокупној сарадњи, корисним саветима и континуираној подршци приликом израде како научног рада, тако и самог мастер рада. Такође захваљујем се и маст. инж. Павлу Васиљевићу на изузетној стручној помоћи и саветима приликом израде овог рада. Њихова ажурност, професионалност и посвећеност у значајној мери су допринеле успешној реализацији овог рада.

Посебно се захваљујем својој породици и пријатељима на константној подршци током целокупног школовања. Такође посебну захвалност дугујем и Дариу на помоћи, стрпљењу, подршци и бодрењу током израде овог мастер рада.

САДРЖАЈ

Списак слика

Списак табела

Скраћенице

PTB-FLA	- *Python Testbed for Federated Learning Algorithms*
MPT-FLA	- *MicroPython Testbed for Federated Learning Algorithms*
ADA	- *Anomaly Detection Algorithm*
IoT	- *Internet Of Things*
I/O	- *Input/Output*
API	- *Application Programming Interface*

1. Увод

Дистрибуирана и децентрализована окружења за федеративно учење омогућавају обуку модела машинског учења уз очување приватности података на хетерогеним уређајима. Једно од таквих решења је PTB-FLA, као и његов пандан заснован на MicroPython (MPT-FLA), који омогућавају експерименталну евалуацију алгоритама федеративног учења у окружењима са различитим хардверским ограничењима. У овом контексту, PFLiForest, федеративни алгоритам за детекцију аномалија заснован на алгоритму изолационих шума (*енгл. Isolation Forest*), имплементиран је коришћењем PTB-FLA. Овај алгоритам омогућава дистрибуирано обучавање модела за детекцију аномалија без потребе за централизованим прикупљањем осетљивих информација, што га чини погодним за примену у окружењима где је приватност података од посебног значаја.

Како би превазишао ограничења истраживања која су му претходила, овај рад, као проширење рада [1] настоји да додатно размотри и испита применљивост постојећих приступа у области федеративне детекције аномалија у реалним системима. У том контексту, рад представља адаптацију наведеног алгоритма за детекцију аномалија за извршавање у MicroPython коришћењем MPT-FLA окружења, циљајући IoT уређаје са ограниченим ресурсима. Предложена адаптација узима у обзир меморијска и рачунарска ограничења својствена уграђеним платформама, при чему се настоји очувати функционалност и логика оригиналне имплементације PFLiForest. Полазећи од тог решења, у овом раду предложени систем је додатно унапређен реализацијом механизма за наизменично обучавање и закључивање.

Предложено решење је експериментално евалуирано у три различита окружења: на једном локалном систему, у дистрибуираном систему са више рачунарских чворова повезаних у локалну мрежу, као и на Raspberry Pi Pico W уређајима који комуницирају у оквиру Wi-Fi мреже. У оквиру експеримента на Raspberry Pi Pico W платформи

коришћени су и додатни температурни сензори, што је омогућило прикупљање реалних података из окружења и додатну евалуацију предложеног приступа у условима блиским стварним IoT системима, укључујући и проверу функционалности механизма са наизменичним режимима обучавања и закључивања. Резултати показују да MicroPython имплементација производи нумерички конзистентне излазе са оригиналном PTB-FLA верзијом, потврђујући функционалну еквиваленцију под ограниченим условима извршавања на Raspberry Pi Pico W уређајима. Поред тога, експериментална евалуација потврдила је да предложени систем омогућава детекцију аномалија над реалним подацима и периодично прилагођавање модела новим условима у окружењу.

Рад је подељен у осам поглавља. Друго поглавље представља теоријске основе за потпуно разумевање овог рада као и преглед истраживања која су му претходила. У трећем поглављу дат је преглед оригиналног алгоритма за детекцију аномалија као и опис његовог прилагођавања за рад у MPT-FLA окружењу. У четвртом поглављу представљена је евалуација ADA-MPT-FLA алгоритма, експериментална евалуација је спроведена у два окружења: на локалном рачунару и у оквиру локалне рачунарске мреже. Пето поглавље посвећено је опису адаптације ADA-MPT-FLA алгоритма на Raspberry Pi Pico W уређајима, као и реализацији система са наизменичним режимима обучавања и закључивања. Док шесто поглавље обухвата експерименталну евалуацију предложеног приступа на Raspberry Pi Pico W уређајима, укључујући и евалуацију система са наизменичним режимима обучавања и закључивања. Поглавље седам износи закључак рада, док је у осмом поглављу дат списак референци које су коришћене за израду рада.

2. Теоријске основе и сродни радови

У овом поглављу представљене су теоријске основе неопходне за разумевање овог рада и споменути и дати кратак преглед сродних радова који се баве сличним истраживањем. За почетак увешћемо појмове као што су федеративно учење и детекција аномалија, након тога ћемо дати преглед окружења коришћених у имплементацији алгоритма федеративног учења. Док ћемо за крај дати преглед сродних радова који су претходили овом.

2.1 Федеративно учење

Федеративно учење [7] представља приступ машинског учења заснован на дистрибуираном обучавању модела уз очување приватности. Процес учења укључује велики број клијената који заједно обучавају модел, али под надзором централног сервера, док подаци за обуку остају децентрализовани. Што значи да сваки клијент обучава модел над сопственим локалним подацима и шаље само ажуриране моделе централном серверу који их агрегира у глобални модел. Овај начин рада је веома добар зато што омогућава додатну приватност, сигурност и права приступа над локалним подацима, јер се ти подаци не централизују нити се деле са сервером. У односу на остале приступе машинског учења, федеративно учење карактерише неколико специфичних својстава: хетерогена расподела података између учесника чворова, при чему локални подаци појединачног чвора не морају бити репрезентативни за целокупан скуп података, неуравнотежена количина локално доступних података за обучавање, ограничени комуникациони ресурси између чворова и сервера, као и могућност масовне дистрибуираности, односно рада у окружењима са великим бројем учесника, који може превазилазити просечан број узорака доступних на појединачном уређају. Претпоставка је да се процес обуке у федеративном учењу одвија када су уређаји у повољном стању, односно када су расположиви ресурси

као што су процесорско време, батерија и мрежна повезаност, при чему се учешће у обуци не одвија континуирано. Ова карактеристика чини федеративно учење погодним за примену у окружењима са уређајима ограничених ресурса на ивици интернета, док се задржавањем података на локалним уређајима унапређује приватност и смањује потреба за њиховом централизацијом, чиме се доприноси и скалабилности система. У пракси, најчешће коришћен приступ представља федеративно усредњавање (*енгл. Federated Averaging*), у оквиру које клијенти локално обучавају модел, након чега се ажурирани параметри шаљу серверу ради агрегације.

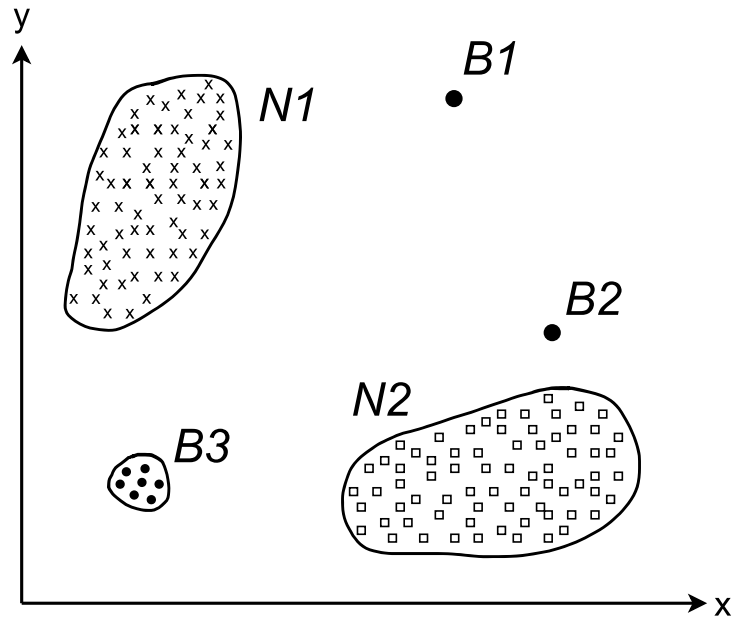
Федеративно учење се може класификовати [8] на хоризонтално федеративно учење, вертикално федеративно учење и федеративно учење преносом знања (*енгл. Federated Transfer Learning*). Код хоризонталног федеративног учења сви учесници располажу подацима исте структуре, односно истим карактеристикама, али сваки учесник поседује сопствени скуп узорака. Код вертикалног федеративног учења учесници располажу подацима о истим објектима или корисницима, али сваки од њих поседује различите карактеристике тих података. Федеративно учење преносом знања примењује се у ситуацијама када се код учесника разликују и врста података и скупови података, при чему знање стечено у једном домену преноси у други ради унапређења процеса обучавања модела. Предложени систем припада хоризонталном федеративном учењу, јер сви клијентски уређаји располажу истом структуром температурних података, док сваки клијент поседује сопствени локални скуп мерења.

2.2 Детекција аномалија

Аномалије [6] су обрасци у подацима који се не уклапају у јасно дефинисан појам нормалног понашања. Веома је важно открити аномалије у подацима због чињенице да се ти подаци врло брзо могу претворити у значајне, а често и критичне, корисне информације у широком спектру домена примене. Неки од значајних примена су аномални обрасци саобраћаја у рачунарској мрежи који може открити да компромитован уређај шаље осетљиве податке на неовлашћено одредиште, док аномалије у подацима о трансакцијама кредитним картицама могу указивати на крађу кредитне картице или идентитета, итд. Због ове практичне важности, проблем детекције аномалија, односно одступање у подацима, предмет је интензивног истраживања током дужег временског периода, што је резултовало развојем великог броја различитих техника и алгоритама за детекцију аномалија.

Ипак, упркос значајном напретку у овој области, поуздано откривање аномалија и даље представља изазован задатак. Основни проблем огледа се у тешкоћи прецизности

дефинисања границе између нормалног и аномалног, јер она често није јасно одређена. Додатне изазове представљају динамичка природа података, различито тумачење аномалија у зависности од домена примене, ограничена доступност означених података за обучавање модела, као и присуство шума који може бити веома сличан стварним аномалијама. Због тога не постоји универзална метода за детекцију аномалија, већ се избор приступа прилагођава карактеристикама података и захтевима конкретне примене.



Слика 1: Аномалије у 2-димензионалном простору

На слици 1 приказан је једноставан дводимензионални скуп података са издвојеним аномалијама. Подаци формирају два нормална региона, означена као N_1 и N_2, с тим да је највећи број посматрања концентрисан унутар ових области. Насупрот томе, тачке које су значајно удаљене од наведених региона, као што су тачке B_1 и B_2, као и тачке које припадају региону B_3, могу се сматрати аномалијама јер одступају од уобичајеног обрасца расподеле података.

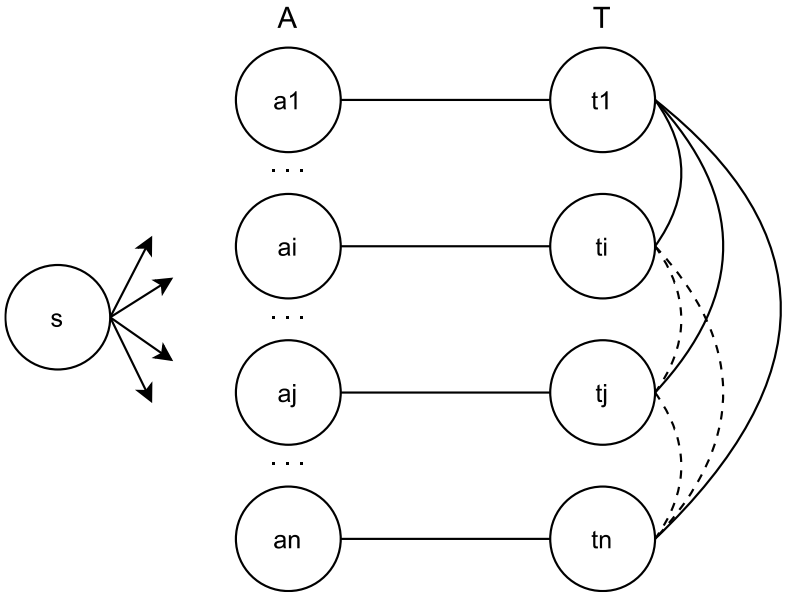
2.3 PTB-FLA

PTB-FLA [3][5] (*енгл. Python Testbed for Federated Learning Alghoritms*) је развијен са примарном намером да се користи као оквир за развој федеративних алгоритама учења или прецизније као окружење за њихово извршавање на једном рачунару. PTB-FLA је писан у чистом Python програмском језику, тако да зависи искључиво само од стандардних Python пакета, као што је multiprocessing. Постоје два разлога зашто је овај начин одабран (i) апликације које се користе у IoT окружењу треба да заузимају минималан простор и (ii) процес инсталације је потребно да буде што једноставнији.

PTB-FLA намеће два ограничења која корисници морају поштовати приликом развоја алгоритама. Прво ограничење подразумева да се развија јединствени апликативни програм, који PTB-FLA окружење накнадно инстанцира и покреће као скуп независних процеса, при чему се понашање сваког процеса одређује на основу његовог идентификатора. Овај захтев произилази из чињенице да је PTB-FLA заснован на шаблону један програм – више података (*енгл. Single program multiple data).* Друго ограничење односи се на чињеницу да корисник имплеметира искључиво функције повратног позива (*енгл. Callback functions*) које описују понашање сервера и клијента, и које се у извршавању позивају од стране генеричког алгоритма федеративног учења унутар PTB-FLA. PTB-FLA подржава и централизоване и децентрализоване алгоритме федеративног учења. Децентрализовани приступ је генерализован тако да сваки процес, односно чвор, може преузети улогу и клијента и сервера.

2.3.1 Архитектура PTB-FLA

Архитектура PTB-FLA (представљена на слици 2), састоји се од процеса за покретање апликације s, дистрибуиране апликације $A = \{a_1, a_2, \ldots, a_n\}$, која представља скуп инстанци апликационог програма a_i, као и дистрибуираног тестног окружења $T = \{t_1, t_2, \ldots, t_n\}$, које представља скуп инстанци тестног окружења t_i, при чему је $i = 1, 2, \ldots, n$, а n означава укупан број инстанци у скуповима A и T.

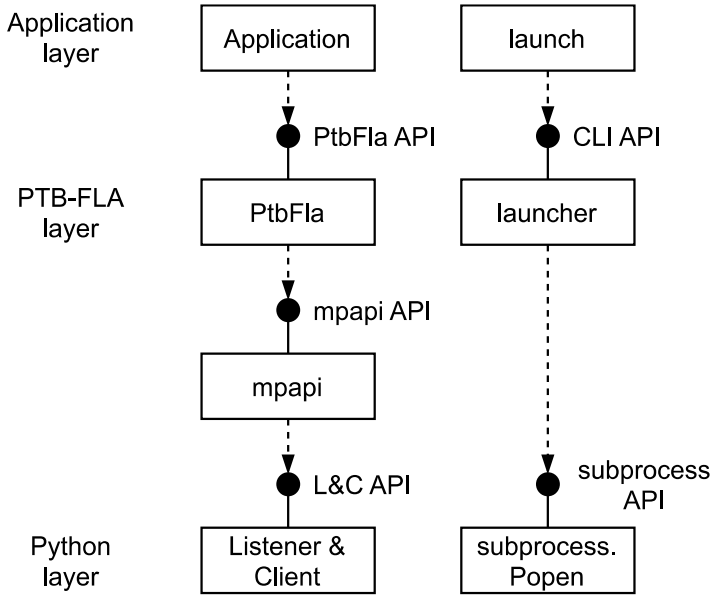


Слика 2: Блок дијаграм архитектуре PTB-FLA
(уз дозволу аутора, преузето из референце [3])

Процес извршавања започиње тако што свака инстанца a_i иницијализује улазне податке на основу аргумената командне линије и потом позива одговарајућу генеричку

API функцију (*fl_centralized* или *fl_decentralized*) нас својом инстанцом окружења t_i. Инстанца окружења на основу идентификатора извршава своју улогу у генеричком алгоритму притом размењујући поруке са другим инстанцама окружења.

Архитектура PTB-FLA система организована је у три слоја која су представљена на слици 3. Највиши слој представља слој дистрибуиране апликације, који обухвата апликативне модуле и пише их корисник. Средњи слој чини PTB-FLA који апстрахује генеричке алгоритме и комуникацију међу чворовима, тј. инстанцама. Најнижи слој представља Python окружење које обезбеђује класе за рад са multiprocessing процесима као и друге Python примитиве.



Слика 3: UML дијаграм архитектуре PTB-FLA
(уз дозволу аутора, преузето из референце [3])

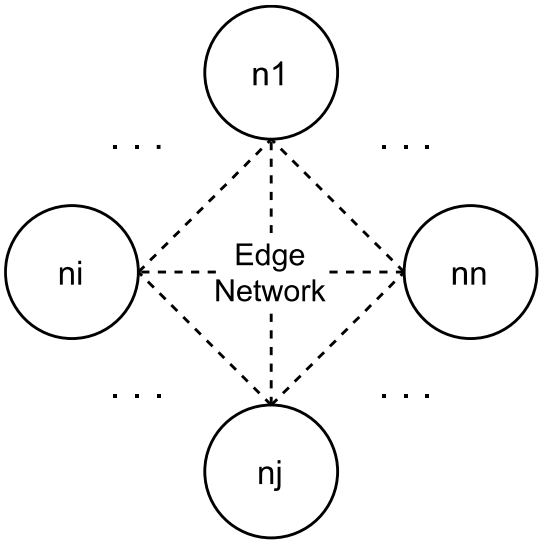
2.4 MPT-FLA

MPT-FLA [4][5] (*енгл. MicroPython Testbed for Federated Learning Algorithms*) представља нови оквир за развој алгоритама федеративног учења који задржава кључне предности PTB-FLA, истовремено превазилазећи његова основна ограничења. За разлику од претходног решења MPT-FLA омогућава покретање појединачних инстанци апликације на различитим мрежним чворовима, укључујући персоналне рачунаре као и IoT уређаје. Пратећи принцип чистог Python као и његов претходник MPT-FLA је заснован на Python асинхроним I/O апстракцијама укључујући асинхроне корутине, токове (*енгл. streams*) и догађаје. MPT-FLA је имплементиран у MicroPython, што представља поједностављену верзију Python3 која се извршава директно на микропроцесорском хардверу без потребе за класичним оперативним системом.

С обзиром на то да MicroPython не подржава Python multiprocessing апстракције које заправо представљају основу PTB-FLA, развој MPT-FLA оквира захтевао је додатно пројектовање постојеће архитектуре и доношење специфичних пројектантских одлука прилагођених ограничењима система на бази микроконтролера. Експериментална валидација у овом окружењу урађена је у бежичној Wi-Fi мрежи коју чине персонални рачунар и Raspberry Pi Pico W уређаји, такође су коришћене апликације преузете из PTB-FLA и прилагођене новом окружењу. Резултати експеримента на крају показују да прилагођене апликације за MPT-FLA окружење показују идентичне нумеричке резултате као и оригиналне апликације извршаване у PTB-FLA окружењу, на тај начин је потврђена тачност прилагођене апликације.

2.4.1 Архитектура MPT-FLA

Систем заснован на MPT-FLA архитектури, представља систем на ивици интернета који се може приказати као граф са *n* чворова, при чему су чворови означени бројевима од 1 до *n,* односно у Python од 0 до *n – 1* и међусобно повезани гранама. При томе чворови представљају процес који се извршава на мањим подсистемима, као што су IoT уређаји, паметни уређаји и персонални рачунари, док гране представљају комуникационе TCP конекције остварене преко интернет мреже. На слици 4 гране су приказане испрекиданим линијама, будући да се одговарајуће комуникационе везе динамички успостављају по потреби, ради размене порука између чворова, потом уклањају.



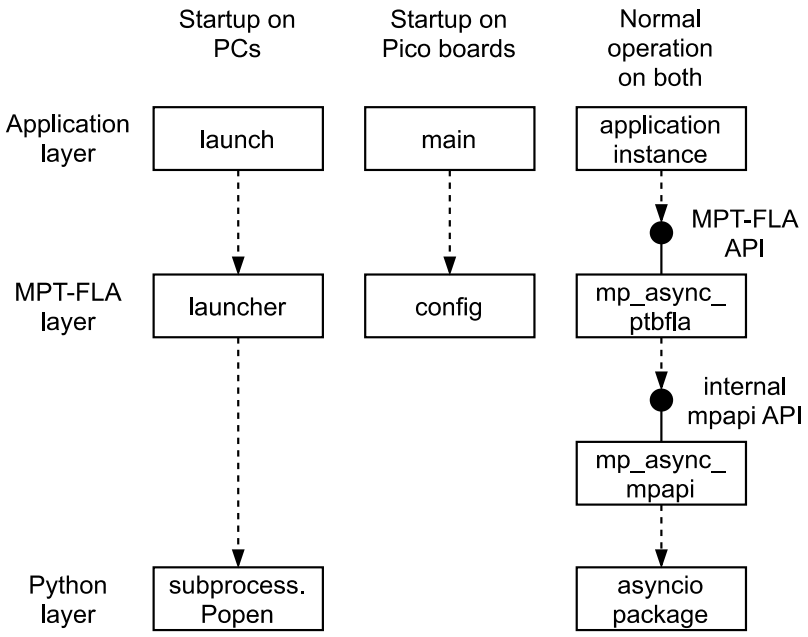
Слика 4: Архитектура MPT-FLA

(уз дозволу аутора, преузето из референце [4])

Приликом иницијализације система, као и током већег дела његовог рада комуникационе везе не постоје, и у току извршавања се по потреби динамички успостављају и раскидају након завршетка њихове употребе. Из тога произилази да ако

два чвора n_i и n_j никада не комуницирају директно између њих неће никада бити успостављена веза, чиме се избегава непотребно коришћење мрежних ресурса. Скуп комуникационих веза које ће бити успостављене зависи од режима рада система тако да и посматрано са становишта топологије мреже структура комуникационе мреже може попримити различите облике. У случају централизованог федеративног учења формира се топологија звезде, при чему централни сервер представља средишњи чвор комуникације. Код децентрализованог федеративног учења формира се потпуно повезан граф, у коме сваки чвор може директно комуницирати са осталим чворовима. Са друге стране, код временског мултиплексирања размена податак узима облик произвољног графа где су само парови чворова повезани. У општем случају сваки чвор n_i извршава једну апликациону инстанцу a_i која на почетку креира одговарајућу инстанцу тестног окружења t_i. Скуп свих апликационих инстанци $\{a_1, \ldots, a_n\}$ чини дистрибуирану апликацију A, док скуп свих инстанци MPT-FLA окружења $\{t_1, \ldots, t_n\}$ чини дистрибуирано окружење T. Током нормалног рада система, дистрибуирана апликација A користи дистрибуирано тестно окружење T за извршавање жељеног алгоритма користећи генеричке алгоритме дате од стране MPT-FLA. Тренутно MPT-FLA подржава три генеричка дистрибуирана алгоритма: (i) централизовани алгоритам федеративног учења, (ii) децентрализовани алгоритам федеративног учења и (iii) алгоритам размене података заснован на временском мултиплексирању.

Архитектура MPT-FLA система организована је хијерархијски и састоји се од три слоја, приказана на слици 5. Највиши слој представља апликациони слој, који обухвата скрипту за покретање на персоналним рачунарима или модул *main* на Raspberry Pi Pico W уређајима, као и апликационе модуле који се користе на обе платформе. Средњи слој чини MPT-FLA слој, који укључује модул *launcher* на персоналним рачунарима, односно *config* на Raspberry Pi Pico W уређајима, као и модуле *mp_async_ptbfla* и *mp_async_mpapi.* Најнижи слој представља Python слој који обезбеђује основне функционалности посредством пакета *asyncio* и *subprocess*. Поред хоризонталне поделе на слојеве, архитектура приказана на слици 5 такође је организована и кроз три вертикалне целине. Лева целина обухвата модуле који се користе приликом покретања система на персоналним рачунарима, средња целина приказује модуле намењене покретању система на Raspberry Pi Pico W уређајима, док десна целина садржи модуле који се користе током нормалног рада система на обе платформе. Архитектура овог окружења може се директно упоредити са архитектуром његовог претходника, при чему су најизраженије промене у средњем слоју. PTB-FLA слој заснован на Python multiprocessing је замењен MPT-FLA слојем који користи *asyncio* пакет као основу за реализацију асинхроног извршавања.



Слика 5: УМЛ дијаграм архитектуре MPT-FLA

(уз дозволу аутора, преузето из референце [4])

При покретању апликациона инстанца користи MPT-FLA API који је имплементиран у оквиру класе PtbFla, у модулу *mp_async_ptbfla*, како би креирала и уништила тестно окружење, иницијализовала чвор, као и извршавала један од три генеричка дистрибуирана алгоритма. Класа PtbFla интерно користи *mpapi* API (*енг. message passing API*), који представља интерни слој за размену података. Овај API обезбеђује скуп од пет корутина заснованих на *asyncio* пакету, које реализују основне механизме размене порука у оквиру система. Корутине *sendMsg* и *RcvMsg* омогућавају слање и пријем појединачних порука између чворова, док се корутина *broadcastMsg* користи за емитовање порука ка свим учесницима у систему. Поред тога, корутина *RcvMsg* омогућава пријем унапред дефинисаног броја порука, обично *n – 1.* Док корутина *server_fun* представља сервисну корутину коју извршава асинхрони задатак креиран приликом покретања система и која координише рад комуникационог слоја. Овакав приступ омогућава кориснику да се фокусира на развој саме апликације федеративног учења. Додатну предност представља подршка за извршавање на микроконтролерима који подржавају MicroPython, где се апликације понашају на еквивалентан начин.

2.5 Сродни радови

У овом одељку дат је преглед радова који се баве сродним темама и представљају основу за развој предложеног решења. Посебна пажња посвећена је радовима из области федеративног учења, детекције аномалија применом изолационих шума, као и имплементација алгоритама машинског учења на уређајима са ограниченим ресурсима.

У последњих неколико година све већа пажња посвећена је примени алгоритма изолационе шуме (*енгл. Isolation Forest*) у оквиру федеративног учења, посебно у системима намењеним IoT окружењима. Васиљевић и сарадници [2] представљају PFLiForest алгоритам за федеративну детекцију аномалија заснован на изолационим шумама у оквиру окружења PTB-FLA. Алгоритам је заснован на формално верификован и структуиран приступ федеративном учењу, чиме се обезбеђује поузданост и коректност процеса обуке уз ефикасно коришћење рачунарских ресурса. Овај рад представља полазну основу за развој ADA алгоритма, чија је адаптација за MPT-FLA окружење предмет овог мастер рада. Рад Xiang и сарадника [9] предлаже адаптацију алгоритма изолационе шуме за федеративно учење увођењем поступка изградње стабла по нивоима, при чему се између клијената и сервера размењују искључиво вредности подела у стаблу, без слања сирових података или целокупних параметара модела. Са сличним циљем, Li [10] предлаже федеративни систем за детекцију аномалија у IoT окружењима, у којем клијенти размењују само метаподатке о чворовима стабала, уз примену Лапласовог шума ради очувања приватности података, на основу тих информација сервер реконструише глобалну изолациону шуму. Ochiai и сарадници [11] представљају дистрибуирани систем за детекцију аномалија на IoT уређајима на ивици интернета, који користи федеративно учење засновано на комуникацији између самих уређаја. Поред обучавања дистрибуираних модела, предложен је и алгоритам за заједничко одређивање глобалног прага аномалије, чиме се омогућава поузданија детекција аномалија на нивоу целог система. Ови радови тематски су блиски алгоритму ADA, јер се заснивају на примени изолационих шума у оквиру федеративног учења за IoT системе. Међутим, у овом раду фокус је стављен не само на адаптацију постојећег ADA алгоритма заснованог на PTB-FLA окружењу [2], за извршавање у MPT-FLA и MicroPython, већ и на његово функционално проширење реализацијом система са наизменичним режимима обучавања и закључивања, као и његову експерименталну евалуацију на уређајима са ограниченим ресурсима.

Поред радова који се баве федеративним учењем, значајан број истраживања посвећен је извршавању алгоритама машинског учења директно на микроконтролерима и другим уређајима ограничених ресурса. Gulati и сарадници [12] представљају TDMiL, оквир који омогућава дистрибуирано обучавање и примену преноса знања *(енгл. transfer learning)* на хетерогеним IoT уређајима заснованим на микроконтролерима, узимајући у обзир ограничења процесорске снаге и меморијског капацитета. Pau и сарадници [13] представљају систем за надзор водоводне мреже који користи алгоритме за детекцију аномалија и континуирано обучавање модела на уређајима на ивици интернета,

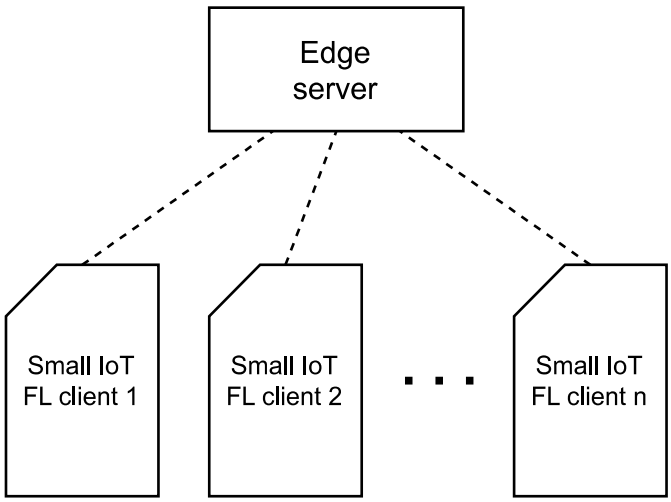
показујући да је обучавање и ажурирање модела могуће реализовати на микроконтролерима. Antonini и сарадници [14] представљају систем за детекцију аномалија у индустријским условима заснован на TinyML приступу. Систем се извршава непосредно на микроконтролеру са MicroPython, при чему уређај самостално прикупља податке, обучава модел заснован на алгоритму изолационе шуме и пријављује детектоване аномалије. Резултати показују да је обучавање и закључивање могуће реализовати директно на уређају са ограниченим ресурсима. Feng и сарадници [15] представљају практично тестно окружење за децентрализовано федеративно учење на уређајима на ивици интернета, наглашавајући значај експерименталне валидације алгоритама на стварном хардверу, а не искључиво у симулираним окружењима.

На основу приказаних радова може се закључити да су постојећа истраживања углавном усмерена или на развој нових алгоритама федеративног учења заснованих на алгоритму изолационих шума, или на примену алгоритама машинског учења на микроконтролерима и другим уређајима ограничених ресурса. За разлику од њих, у овом раду није извршена само адаптација постојећег ADA алгоритма заснованог на PTB-FLA окружењу, за MPT-FLA и MicroPython, већ је реализовано и његово функционално проширење увођењем наизменичних режима обучавања и закључивања, као и његова експериментална евалуација на локалном рачунару, локалној мрежи и Raspberry Pi Pico W уређају.

3. Алгоритам за детекцију аномалија

3.1 ADA за PTB-FLA

Архитектура целог система заснована је на концепту централизованог федеративног учења. Систем се састоји од централног чвора који има улогу координатора процеса учења, као и великог броја клијената који учествују у обучавању модела. Клијентски чворови су имплементирани на малим IoT уређајима опремљеним сензором температуре, док се сервер налази на мрежном ивичном слоју. Архитектура система заснованог на ADA [2], намењеног за детекцију аномалија уз помоћ федеративног учења приказана је на слици 6. Целокупан систем је пројектован са примарним циљем предвиђања температурних аномалија у једнодимензионалном простору, што је важно напоменути.



Слика 6: ADA архитектура система
(уз дозволу аутора преузето из референце [2])

ADA је заснован на PTB-FLA, имплементација се ослања на генерички централизован алгоритам федеративног учења (*fl_centralized*), при чему се примењује принцип један програм више података. Овај принцип подразумева да се иста програмска логика извршава на свим чворовима, тј. да сервер или клијент могу да заузму обе улоге, у зависности од података који су му доступни. Такође у оквиру система сваки чвор понаособ извршава инстанцу PTB-FLA, која је задужена за комуникацију међу чворовима.

Имплементација ADA алгоритма се заснива на процесу прављења појединачних изолационих стабала на федеративни начин, користећи ред како би се пратио раст стабала, уместо ослањања на стек, на тај начин је задржана компатибилност са PTB-FLA окружењем. На почетку прављења изолационих стабала долази до иницијализације корена стабла као и реда који се користи за праћење стабла. Након неопходне иницијализације почиње итеративно извршавање алгоритма где се из реда узима чвор над којим се одређује вредност тачке поделе, све док се ред не испразни у потпуности. Одмах по завршетку израчунавања тачке поделе, долази до синхронизације између свих чворова. Механизам синхронизације је неопходан како би се спречило да неки чворови превремено заврше изградњу тренутног стабла и започну следећу итерацију.

Такође је веома битно поменути како се врши комуникација између сервера и клијента. На почетку, сви клијенти шаљу своје параметре модела, односно вредности тачака поделе изолационог стабла, серверу. Сервер обрађује добијене параметре модела и враћа назад клијенту агрегиране параметре модела, које клијент користи за прављење изолационог стабла. Након тога комуникација између клијента и сервера се одвија још једном, преносећи информације о фази у којој се налази клијент и сервер, у зависности од добијене информације алгоритам се завршава, тј. престаје прављење стабла или прелази у фазу мировања. Клијентски чворови који се налазе у фази мировања директно прелазе у следећу итерацију прављења стабла, све док се не задовоље услови да је достигнута максимална дубина, да је скуп података постао хомоген или да се ред испразнио. Када се сви услови испуне фаза мировања се може сматрати завршетком алгоритма.

Резултат извршеног алгоритма је обучен модел изолационе шуме у облику листе изолационих стабала. Добијен модел је спреман за процес предвиђања аномалија на сваком од клијената и представља резултат заједничког процеса федеративног учења, без дељених приватних података.

3.2 ADA MPT-FLA

Полазећи од оригиналног алгоритма за детекцију аномалија, развијеног у PTB-FLA окружењу, извршена је његова адаптација за рад у MPT-FLA оквиру, узимајући у обзир

ограничења система базираног на микроконтролерима. Први корак ка прилагођавању постојећег алгоритма односио се на увођење асинхроних I/O апстракција, у складу са програмским моделом који подржава MicroPython. Асинхроне I/O апстракције представљају програмски концепт који омогућава паралелно извршавање више задатака без потребе за вишеструким нитима или процесима. У MicroPython, се ове апстракције реализују коришћењем корутина, односно await израза, петље за обраду догађаја (*енгл. event loop*) и механизама за управљање задацима (*енгл. task*), имплементираних у оквиру asyncio модула. Корутине представљају функције које се могу паузирати и наставити, чиме се омогућава ефикасно управљање улазно/излазним операцијама, док петља за обраду догађаја координира њиховим извршавањем у неблокирајућем режиму. У MicroPython, функција се декларише као асинхрона користећи синтаксу async def уместо регуларне def. Ова на изглед мала промена има дубоке импликације на начин извршавања кода. Функција async def позната и као корутина, не извршава се као регуларна синхрона функција, уместо тога она враћа објекат корутине, који се може покренути и контролисати у оквиру петље за обраду догађаја. За активирање, односно позивање, корутине користи се кључна реч await, која омогућава неблокирајућа извршавања улазно/излазних операција.

На основу претходно описаних асинхроних апстракција, оригинални алгоритам је модификован тако да користи async функције и await изразе за неблокирајуће извршавање задатака у оквиру MPT-FLA окружења.

Табела 1: Конверзија метода у асинхроне корутине и њихова интеграција применом *await* израза

Методе конвертоване у асинхроне корутине
01: async def build_isolation_tree(ptb: PtbFla, data, max_depth=10) 02: async def build_isolation_forest(ptb: PtbFla, data, num_trees=10, max_depth=10) 03: async def benchmarkForest(ptb: PtbFla,sessionName, params, numberOfIterations=1) 04: async def testing_main()
Методе које захтевају await
01: ret = await ptb.fl_centralized(02: fl_cent_server_processing, 03: fl_cent_client_processing, 04: lData, 05: pData, 06: 1,) 07: ph = await ptb.fl_centralized(08: fl_cent_server_phase, 09: fl_cent_client_phase, 10: ret["phase"], 11: ret["phase"], 12: 1,) 13: await build_isolation_tree (ptb, data, max_depth)

```
14: await build_isolation_forest(ptb, data, num_trees, max_depth)
15: await benchmarkForest(ptb, sessionName, params, numberOfIterations=20)
16: await ptb.start()
```

Док је део функционалности захтевао структурну конверзију у асинхрон облик ради усклађивања са програмским моделом MicroPython окружења, функције повезане са комуникационим и синхронизационим механизмом морале су експлицитно користити await. На тај начин се обезбеђује правилан редослед извршавања у дистрибуираном окружењу.

Табела 2: Покретање програма путем *asyncio event loop*

Покретање главног програма коришћењем async event loop
01: def run_testing_main(): 02: asyncio.run(testing_main())

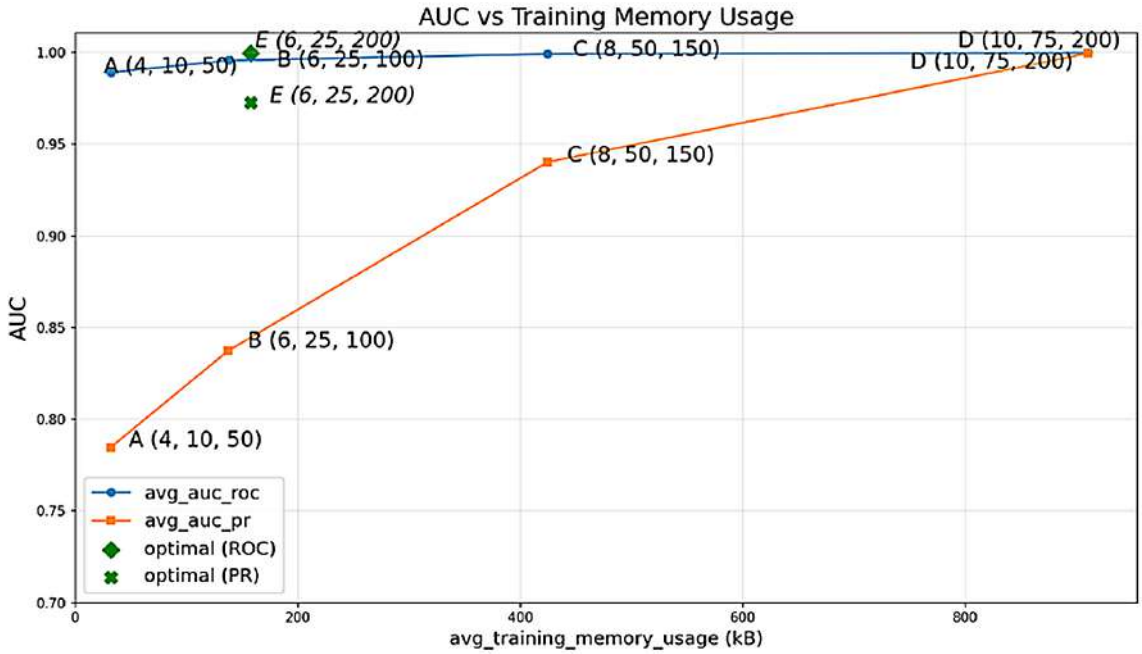
Како би се омогућило извршавање асинхроних корутина, целокупан ток програма покреће се кроз *asyncio* петљу за обраду догађаја (енгл. *asyncio event loop*). За разлику од стандардног синхроног позива главне функције, у овом приступу извршавање започиње позивом функције asyncio.run(), која управља распоређивањем и извршавањем свих асинхроних задатака унутар програма.

4. ADA-MPT-FLA евалуација

У овом одељку приказани су резултати експерименталне евалуације ADA за MPT-FLA. Скуп података коришћен за мерење перформанси представља скуп вредности температуре изражених у степенима Целзијуса. Перформансе модела су добијене коришћењем следећих метрика: површина испод ROC криве (*енгл. Receiver Operating Characteristic*) и површина испод криве прецизности и осетљивости (*енгл. Precision-Recall, PR*)

AUC-ROC (*енгл. Area Under Curve Receiver Operating Characteristic*) представља квантификацију целе ROC криве, где вредност површине испод криве представља укупну дискриминаторну моћ модела у разликовању нормалних и нерегуларних случајева. Вредност која је ближа 1 означава висококвалитетан модел, док вредност 0.5 имплицира да модел нема дискриминаторну моћ и да је еквивалент случајном погађању.

AUC-PR (*енгл. Area Under Curve Precision Recall*) представља квантификацију целе PR криве, где ова метрика представља просечну перформансу модела у одржавању прецизности при различитим нивоима одзива. Високе вредности ове метрике указују да модел истовремено задржава високе вредности прецизности и осетљивости. У поређењу са AUC-ROC, ова метрика је стабилнија и информативнија у сценаријима са неуравнотеженим подацима.



Слика 7: Оптимална тачка E

(уз дозволу аутора преузето из [2])

У односу на остале тачке које су приказане на слици 7, тачка E је емпиријски изабрана из скупа свих експерименталних конфигурација. На основу вредности AUC-ROC и AUC-PR изабрана тачка E приказује веома добре перформансе модела, због чега је усвојена као референтна тачка за даљу експерименталну евалуацију.

4.1 На локалном рачунару

Експериментална евалуација ADA-MPT-FLA алгоритма на локалном рачунару је изведена у контролисаном окружењу у коме су серверске и клијентске инстанце апликације покренуте као одвојени процеси на истом рачунару. Евалуација ADA-MPT-FLA алгоритма заснована је на експериментима са следећим вредностима параметара: величина изолационе шуме је 25 стабала, максимална дубина стабла је 6, док је величина скупа података за обуку 200. За изабрану тачку E (6, 25, 200) вредност за AUC-ROC је 0.99999 и вредност AUC-PR је 0.99994. Ове вредности су добијене понављањем експеримента 40 пута, како би се варијабилност свела на минимум уз задржавање идентичних скупова података за обуку из итерације у итерацију.

4.2 На локалној рачунарској мрежи

Експериментална евалуација ADA-MPT-FLA алгоритма на локалној рачунарској мрежи спроведена је са циљем валидације понашања алгоритма у реалним дистрибуираним системима. Експериментална поставка била је сачињена од укупно три

рачунара у оквиру исте локалне мреже, при чему је један рачунар био серверски чвор, док друга два рачунара су били клијентски чворови. Како би се омогућило прикупљање података са оба клијентска чвора на серверском рачунару направљен је један дељени директоријум где су клијентски чворови могли да приступе и уписују своје податке у базу података након сваке итерације. Подаци се записују након сваке обраде, при чему овај процес не утиче на мерења везана за време извршавања. На овај начин је обезбеђена централизована евиденција резултата, као и синхронизован приступ подацима између серверског и клијентског чвора без потребе за додатним мрежним сервисима, што додатно олакшава накнадну обраду података.

Као и у претходној експерименталној евалуацији на једном рачунару, изабрана је тачка E (6, 25, 200) која даје 0.99977 вредност за AUC-ROC и 0.99242 вредност за AUC-PR. Такође су и ове вредности добијене понављањем експеримента 40 пута, уз задржавање идентичних скупова података из итерације у итерацију.

Изабрана тачка E (6, 25, 200) представља оптималну тачку јер је показала добру равнотежу између перформанси модела у смислу AUC-ROC и AUC-PR. Добијене вредности AUC-ROC и AUC-PR остају скоро непромењене, без обзира да ли је извршавање било на једном рачунару или на локалној рачунарској мрежи. Експериментална евалуација потврђује да ADA-MPT-FLA имплементација даје нумерички конзистентне резултате у односу на оригинални ADA-PTB-FLA алгоритам.

5. ADA-MPT-FLA за Raspberry Pi Pico W

Приликом покретања прилагођеног ADA-MPT-FLA алгоритма на Raspberry Pi Pico W уређај, уочено је да није било довољно увести само асинхроне I/O апстракције. Поред тога, била је неопходна и додатна модификација самог алгоритма како би се прилагодио ограничењима MicroPython и расположивим хардверским ресурсима. У наставку су приказане кључне измене које су извршене у имплементацији алгоритма у циљу његовог прилагођавања MicroPython:

- Сви резултати појединачног извршавања алгоритма чувани су у бази података, након чега су агрегирани ради добијања коначних резултата. С обзиром на то да MicroPython не пружа подршку за рад са базама података, ова функционалност је замењена радом са текстуалним датотекама, што омогућава складиштење и накнадну обраду података на оваквом систему. Како би се ове текстуалне датотеке пребациле на серверску страну направљена је скрипта која по покретању, шаље команду путем серијске везе клијентској страни. Путем серијске везе шаљу се подаци из клијентске текстуалне датотеке и записују у одговарајућу текстуалну датотеку на серверској страни.
- У поређењу са стандардним Python који пружа велики скуп библиотека за коришћење, MicroPython је много више ограничен што значи да обезбеђује много мањи подскуп стандардних библиотека. Због ограничења скупа доступних стандардних библиотека није било могуће користити библиотеку Numpy. Уместо тога, коришћена је библиотека ulab која представља имплементацију ограниченог подскупа функционалности библиотеке Numpy, прилагођеног извршавању у MicroPython. Поред изостанка подршке за библиотеку Numpy, MicroPython не подржава ни библиотеке као што су

pandas, time, tracemalloc и tqdm, док се имплементација библиотеке random делимично разликује у односу на стандардни Python.

- Због ограничења меморијских ресурса и начина обраде података у MicroPython, било је потребно смањити број одбирака који се чувају у текстуалним датотекама. Првобитан број одбирака од 10000 није могао бити обрађен на циљаном уређају, због тога је број одбирака смањен на вредност која представља 600 одбирака, што омогућава стабилно извршавање програма. Број одбирака је смањен јер није било могуће доделити довољно меморије за првобитан број одбирака.
- Још једна неопходна измена која је била потребна због недостатка меморије односила се на коришћење у потпуности Python библиотеке. Конкретно за евалуацију перформанси алгоритма коришћене су функције из библиотеке sklearn.metrics. Пошто MicroPython не подржава ту библиотеку, одређене функције за евалуацију модела као што су: roc_auc_score(), auc() и precision_recall_curve() нису могле бити директно коришћене на уређајима. Због тога је евалуација перформанси модела у потпуности извршавана на рачунару са пуном Python имплементацијом, тј. на серверском чвору.

Поред измена у коду које су се односиле на сам алгоритам било је потребно додати и две конфигурационе датотеке: config.py и main.py како би се омогућила основна комуникација као и иницијализација и покретање самог алгоритма. Датотека config.py служи за специфичну конфигурацију параметара окружења као и мрежних подешавања, док main.py обезбеђује иницијализацију и покретање самог алгоритма. Config.py омогућава да се поставе следећи параметри: мрежни подаци, IP адресе чворова, као и броја чворова. Мрежни подаци обухватају SSID и лозинку саме Wi-Fi мреже, IP адреса чворова представља IP адресу персоналног рачунара на ком се налази серверски чвор, док параметар броја чворова дефинише укупан број чворова који учествују у систему. Main.py поред основне функционалности иницијално успоставља повезивање чворова на мрежу, чиме се омогућава комуникација и размена података са осталим чворовима у систему.

Извршене модификације нису утицале на логику самог алгоритма, већ су биле усмерене на прилагођавање извршавања програма на MicroPython. На овај начин је очувана функционална еквивалентност у поређењу са оригиналном имплементацијом.

5.1 Имплементација иницијалног система за детекцију аномалија

Поред адаптације ADA-MPT-FLA алгоритма за извршавање у MicroPython, било је неопходно извршити и додатне измене како би се омогућио развој целокупног система за детекцију аномалија на Raspberry Pi Pico W уређајима. За разлику од претходне имплементације, која је била усмерена искључиво на процес обучавања модела, предложено решење омогућава континуиран рад система кроз наизменично извршавање режима обучавања и режима закључивања. На овај начин систем је у могућности да периодично прилагођава модел новоприкупљеним подацима, а између две фазе обучавања користи последњи обучени модел за детекцију аномалија над подацима који се прикупљају у реалном времену. У наставку су описане кључне измене које су извршене како би се омогућила реализација оваквог начина рада, при чему су измене биле усмерене на организацију рада система, управљање процесом обучавања и закључивања, као и на прикупљање, складиштење и обраду података у условима ограничених хардверских ресурса.

Први корак у реализацији предложеног система односио се на омогућавање закључивања над реалним подацима који се прикупљају са температурног сензора који је који је повезан на Raspberry Pi Pico W уређајем. Претходна имплементација ADA-MPT-FLA алгоритма била је усмерена искључиво на процес обучавања модела, при чему се обучени модел налазио искључиво у радној меморији током извршавања програма. Такав приступ није омогућавао његову поновну употребу након завршетка обучавања, нити детекцију аномалија над новим подацима без поновног покретања целокупног процеса обучавања.

Како би се овај недостатак превазишао, уведен је механизам за трајно чување обученог модела. Приликом завршетка процеса обучавања модела, добијени модел се записује и JSON датотеку, из које се по потреби поново учитава у радну меморију. На овај начин је омогућено коришћење једном обученог модела у процесу закључивања, без потребе за поновним покретањем целокупног процеса обучавања. С обзиром на то да сваки клијент у систему располаже сопственим локалним моделом, за сваки клијентски чвор формира се посебна JSON датотека у којој је сачуван одговарајући обучен модел. Приликом покретања процеса закључивања сваки клијент учитава сопствени модел из припадајуће JSON датотеке и користи га за детекцију аномалија над температурним подацима који се у реалном времену прикупљају са локалног сензора. Након учитавања модела из JSON датотеке, свака нова вредност температуре очитана са сензора користи се

као улаз у процес закључивања. За сваки очитани узорак израчунава се мера аномалности, након чега се врши њено поређење са оптималним прагом одлучивања. На основу резултата овог поређења доноси се одлука да ли узорак представља аномалију. Као визуелна индикација резултата детекције, на Raspberry Pi Pico W уређају користи се LED диода. У случају да је очитана температура класификована као аномалија, LED диода се укључује, док у супротном остаје искључена. Ради лакшег праћења система и накнадне анализе резултата, имплементиран је и механизам за евидентирање резултата детекције у лог датотеку. За свако очитавање бележи се вредност температуре, као и информације о томе да ли је узорак класификован као аномалија или као нормално стање. На овај начин омогућено је једноставније праћење понашања система током рада, док су резултати добијени овим мерењима приказани у поглављу 6.1, посвећеном експерименталној евалуацији алгоритма на Raspberry Pi Pico W уређају.

5.2 Реализација система са наизменичним обучавањем и закључивањем

Након реализације процеса закључивања било је неопходно омогућити и континуирано поновно обучавање модела. Процес обучавања започиње провером постојања претходно обученог модела. Приликом покретања система сваки клијентски чвор прво провери да ли за њега већ постоји JSON датотека у којој је сачуван иницијални модел. Уколико таква датотека не постоји, клијент најпре покреће процес обучавања над иницијалним локалним подацима, након чега се добијени модел чува у одговарајућој JSON датотеци. Тек након формирања иницијалног модела, систем може да настави са процесом закључивања и каснијег поновног обучавања на новоприкупљеним подацима. На овај начин избегава се непотребно поновно формирање иницијалног модела приликом сваког покретања система, чиме се смањује време потребно за иницијализацију и омогућава несметан наставак рада са претходно обученим моделом. Након успешне иницијализације и учитавања постојећег модела, систем наставља са прикупљањем нових података са температурног сензора, који представљају основу за процес поновног обучавања.

Како би се омогућило поновно обучавање модела на актуелним подацима из окружења, било је неопходно обезбедити континуирано ажурирање локалног скупа података сваког клијента. Из тог разлога свака нова вредност температуре очитана са сензора користи се за ажурирање локалног скупа података који представља улаз у процес обучавања. Локални скуп података сваког клијента складишти се у текстуалној датотеци фиксне величине. Приликом сваког новог очитавања температуре, нова вредност се

уписује на случајно одабрану позицију у постојећој датотеци, чиме се једна од претходно сачуваних вредности замењује новом. На овај начин обезбеђује се постепено освежавање скупа података без повећања његове величине, што је од посебног значаја у условима ограничених меморијских ресурса Raspberry Pi Pico W уређаја. Истовремено, локални скуп података континуирано се прилагођава новим мерењима, чиме се омогућава да процес обучавања користи најактуелније информације о стању околине. Након ажурирања локалног скупа података, над истом очитаном вредношћу температуре извршава се процес закључивања коришћењем тренутног модела, чиме се одређује да ли посматрани узорак представља аномалију. На овај начин свака новоприкупљена вредност истовремено доприноси ажурирању скупа података за будуће обучавање и омогућава непосредну детекцију аномалија.

Процес поновног обучавања не покреће се након сваког новог очитавања температуре, већ тек након што се прикупи унапред дефинисан број нових узорака. У предложеној имплементацији овај праг одговара ажурирању 25% локалног скупа података, чиме се обезбеђује да скуп буде довољно ажуриран пре покретања новог циклуса обучавања. По достизању овог услова започиње процес прављења новог стабла изолационе шуме, при чему се као улаз користи ажурирани локални скуп података клијената. Добијено стабло се затим користи за ажурирање постојеће изолационе шуме, без потребе за поновним формирањем целокупног модела. Ажурирање се реализује заменом једног од постојећих стабала у изолационој шуми новоизграђеним стаблом. Број стабала који чине изолациону шуму дефинише се приликом иницијалног обучавања модела и остаје непромењен током рада система. Новоизграђено стабло не додаје се постојећем моделу, већ замењује једно од стабала која већ припадају изолационој шуми. Свакој вредности индекса ажурирања одговара једно стабло у изолационој шуми, тако да новоизграђено стабло увек заузима место стабла чији је редни број једнак тренутној вредности индекса. Након сваког ажурирања вредност индекса се увећава за један, тако да се приликом наредног циклуса обучавања ажурира следеће стабло у изолационој шуми. Након што се ажурира последње стабло у изолационој шуми, вредност индекса се враћа на почетак, чиме се обезбеђује циклично ажурирање свих стабала изолационе шуме**.** Оваквим приступом избегава се поновна изградња целокупне изолационе шуме, чиме се смањују меморијски и процесорски захтеви током поновног обучавања, што је од посебног значаја у условима ограничених ресурса Raspberry Pi Pico W уређаја.

Ради лакшег разумевања имплементације, у табели 3 дат је преглед најзначајнијих функција које реализују предложени систем. У наставку поглавља детаљно су описане функције које представљају кључне делове имплементације.

Табела 3: Преглед кључних функција имплементираног система

Функција	Опис
get_temperature()	Очитава температуру са уграђеног температурног сензора.
update_dataset_with_sensor_data()	Ажурира локални скуп података са новим температурама са сензора.
check_temperature_anomaly()	Обавља процес закључивања и утврђује да ли очитана температура представља аномалију.
make_one_tree()	Формира једно ново стабло на основу ажурираног локалног скупа података.
update_forest()	Ажурира постојећу изолациону шуму заменом једног стабла новоизграђеним стаблом.

Описаним изменама реализован је систем који омогућава континуирано прикупљање података са температурног сензора, детекцију аномалија у реалном времену и периодично ажурирање модела на основу новоприкупљених података. На овај начин омогућен је непрекидан рад система, при чему се процеси обучавања и закључивања извршавају наизменично, уз прилагођеност ограниченим меморијским и процесорским ресурсима Raspberry Pi Pico W уређаја.

6. ADA-MPT-FLA-RPi евалуација

Експериментална евалуација ADA-MPT-FLA алгоритма извршена је на Raspberry Pi Pico W плочи како би се верификовала функционалност адаптираног алгоритма. Експериментална валидација ADA-MPT-FLA алгоритма је извршена на Wi-Fi мрежи, која обухвата бежични рутер, две Raspberry Pi Pico W плоче и један персонални рачунар. У овој поставци персонални рачунар је имао улогу серверског чвора, док су два Raspberry Pi Pico W уређаја имала улогу клијентских чворова. Целокупна комуникација између чворова одвијала се преко бежичне мреже путем Wi-Fi рутера који је обезбеђивао мрежну повезаност унутар локалне мреже. Како би се уређај повезао на мрежу потребно је исправно конфигурисати датотеку config.py. Исправна додела конфигурационе датотеке сваком уређају је кључна, јер параметри морају одговарати улози уређаја у експерименту. На пример, конфигурација намењена за Raspberry Pi Pico W уређају са идентификатором 1 мора садржати његов јединствен идентификатор, као и IP адресу серверског чвора. Аналогно томе, конфигурациона датотека уређаја са идентификатором 2 мора бити подешена са одговарајућим идентификатором чвора и истом IP адресом серверског чвора. Исправна конфигурација ових параметара представља предуслов за успостављање мрежне комуникације између чворова. У супротном, комуникација неће бити успешно успостављена, што онемогућава извршавање самог алгоритма.

Као и у претходне две фазе евалуације, почетна тачка анализе била је тачка E (6, 25, 200), при чему су добијени слични резултати перформанси, уз укупно време обучавања модела од приближно 28 минута по итерацији. Продужено време обучавања, као и уочени прекиди у обучавању који су највероватније последица нестабилности мреже, указали су на ограничењима у погледу практичне применљивости ове конфигурације.

Након тога, тачка E’ (6, 15, 200) изабрана је као лакша алтернатива у погледу потрошње меморије и укупног времена обучавања, при чему су постигнуте вредности

AUC-ROC од 0.95262 и AUC-PR од 0.78426. С обзиром на повећано време извршавања по итерацији на Raspberry Pi Pico W уређају, број итерација је редукован са 40 на 20.

Додатно је узета у обзир и тачка E” (6, 10, 200), при чему су постигнуте вредности AUC-ROC од 0.94745 и AUC-PR од 0.75749. У Табели 4 приказано је поређење перформанси модела у односу на AUC-ROC и AUC-PR за одабране конфигурације параметара. Такође, треба напоменути да су резултати перформанси за тачку E преузети из одељка 4.2, јер су поновљени прекиди у обучавању могли утицати на поузданост израчунатих процена перформанси.

На Raspberry Pi Pico W уређају просечно време извршавања једне итерације је приближно 14 минута, док је укупно време извршавања за 20 итерација било око 5 сати и 30 минута. Одговарајућа времена извршавања за тачке E’ и E” приказане су у Табели 4. У поређењу са извршавањем на PC платформи, где је целокупно време извршавања за 20 итерација износило свега неколико минута, уочава се да алгоритам захтева знатно више времена на Raspberry Pi Pico W уређају, што указује на ограничења уређаја са ограниченим ресурсима. За тачку E” (6, 10, 200) забележено је значајно смањење времена извршавања у односу на претходну конфигурацију параметара, при чему је укупно време за 20 итерација износило 2 сата и 30 минута, уз приближно 8 минута по итерацији.

Табела 4: Поређење времена извршавања за одабране конфигурације параметара

Метрика	Тачка E	Тачка E’	Тачка E’	Тачка E’’	Тачка E’’
Платформа	PC	PC	Raspberry Pi Pico W	PC	Raspberry Pi Pico W
Параметри	(6, 25, 200)	(6, 15, 200)	(6, 15, 200)	(6, 10, 200)	(6, 10, 200)
AUC-ROC	0.9999	0.9996	0.9526	0.9989	0.9474
AUC-PR	0.9999	0.9699	0.7843	0.9259	0.7575
Време по итерацији	~ 27 s	~ 6 s	~ 14 min	~ 4 s	~ 8 min
Број итерација	20	20	20	20	20
Укупно време	9 min	2 min	5 h 30 min	1.5 min	2 h 30 min

Уочено смањење времена извршавања је последица измене другог параметра, који представља број стабала у алгоритму и непосредно утиче на сложеност алгоритма и на време његовог извршавања. У оквиру спроведене експерименталне поставке као гранична вредност за број стабала узет је број 10, будући да представља минималан праг за добијање смислене и поуздане процене перформанси модела.

Иако је време извршавања самог алгоритма на Raspberry Pi Pico W уређају значајно дуже у поређењу са PC уређајем, алгоритам се успешно извршио и применљив је на систему са ограниченим ресурсима.

6.1 Евалуација иницијалног система за детекцију аномалија

У оквиру експерименталне евалуације иницијалног система за детекцију аномалија извршена је провера исправности рада иницијално обученог модела приликом детекције аномалија над температурама очитаним са уграђеног температурног сензора Raspberry Pi Pico W уређаја. Циљ експеримента био је да се испита способност модела да разликује нормалне температурне вредности од наглих промена температуре настале услед спољашњих утицаја.

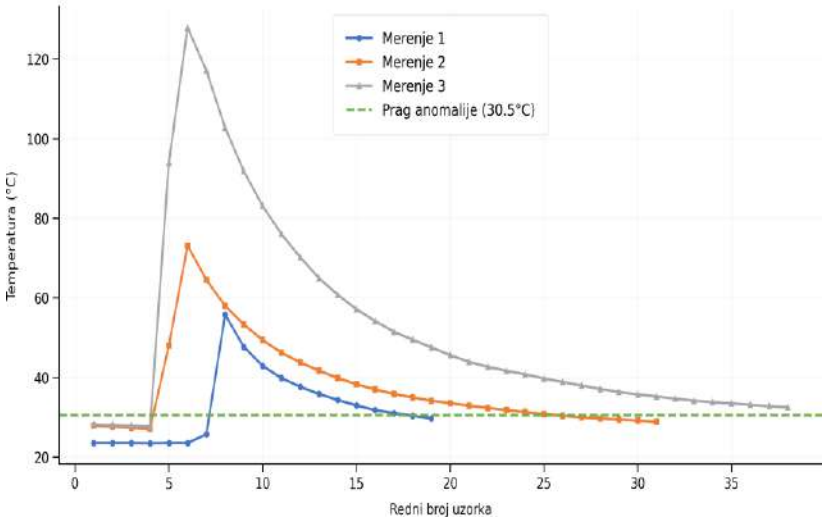
Као у претходним фазама евалуације полазна тачка анализе била је тачка E', која одговара конфигурацији параметара (6, 15, 200). Наведена конфигурација подразумева максималну дубину стабла од 6, изолациону шуму са 15 стабала и локални скуп података од 200 узорака. За доношење одлуке о присуству аномалије коришћен је оптимални праг одлучивања вредности 0.81, који је одређен као вредност за коју F1 мера достиже максимум на валидационом скупу података. У циљу провере исправности рада иницијално имплементираног система спроведене су две групе експеримената. Прва група експеримената обухватала је загревање температурног сензора коришћењем упаљача, чиме су симулиране нагле промене температуре. Друга група експеримената реализована је коришћењем уређаја који обезбеђује континуирани ток топлог ваздуха , при чему су мерења извршена на три различите удаљености од сензора (30cm, 20cm, 10cm), како би се испитало понашање система при различитим интезитетима загревања. Током сваког експеримента бележене су очитане температуре и резултат детекције аномалије. На основу података забележених у лог датотекама урађена је анализа понашања система, а сажет преглед добијених резултата приказан је у табели 5.

Табела 5: Сажет преглед резултата евалуације иницијалног система за детекцију аномалија

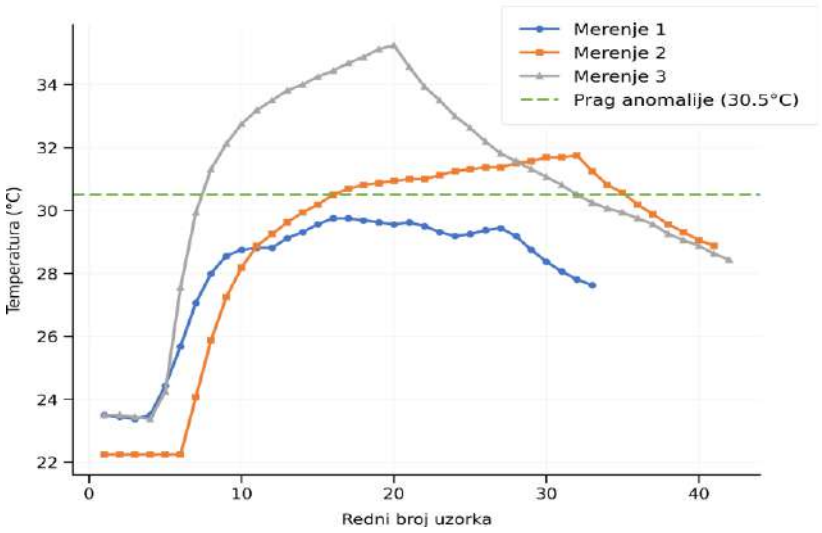
Експеримент	Начин изазивања промене температуре	Максимална температура [°C]	Понашање система
E1	Нагло загревање сензора	55,81	Аномалија успешно детектована
E2	Нагло загревање сензора	73,13	Аномалија успешно детектована
E3	Нагло загревање сензора	127,94	Аномалија успешно

			детектована
E4	Континуирани извор топлог ваздуха (30cm)	29,75	Аномалија није детектована
E5	Континуирани извор топлог ваздуха (20cm)	31,75	Аномалија успешно детектована
E6	Континуирани извор топлог ваздуха (10cm)	35,25	Аномалија успешно детектована

Ради лакшег увида у промену температуре током експеримената, у наставку су приказани и графички прикази карактеристичних мерења.



Слика 8: Детекција аномалије при наглом порасту температуре



Слика 9: Детекција аномалије при постепеном загревању температуре

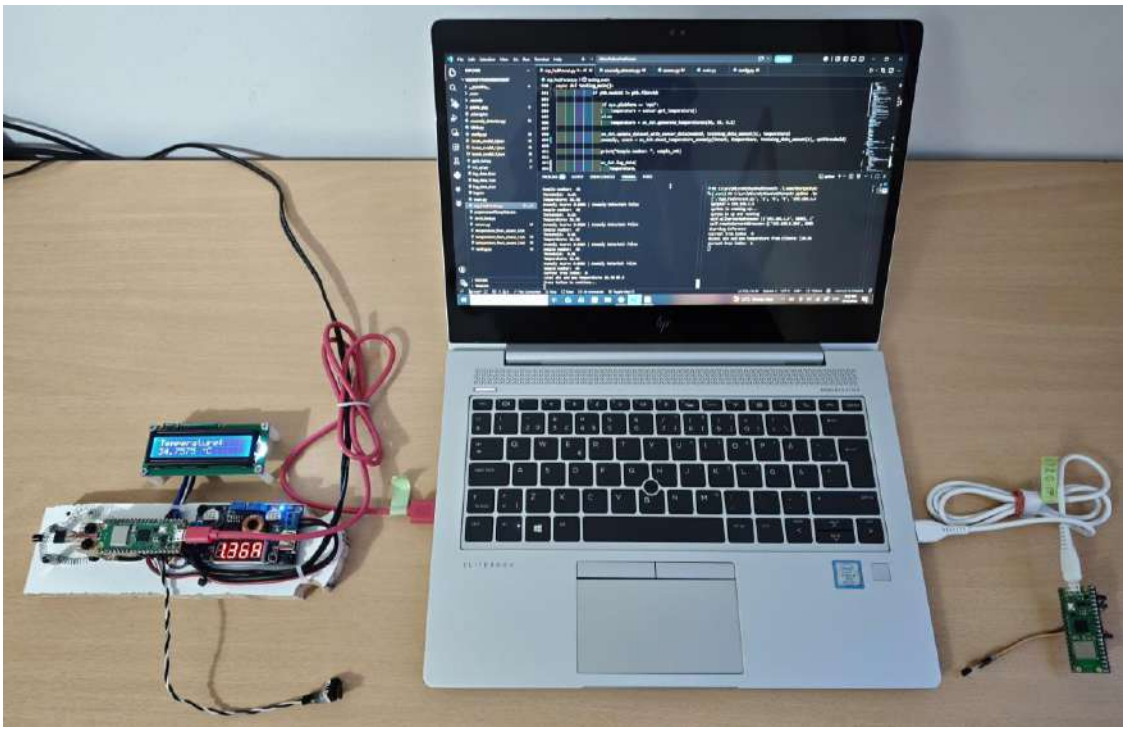
Резултати приказани у табели 5 и на слици 8 и слици 9 показују да иницијално имплементирани систем успешно детектује аномалије настале као последица наглих промена температуре. У свим експериментима који су подразумевали значајно повећање температуре сензора аномалија је успешно детектована, што потврђује исправност

процеса закључивања над реалним подацима. Поред тога, резултат показују да систем није ограничен искључиво на нагле промене температура, већ успешно препознаје и постепено загревање сензора када оно доведе до довољног великог одступања од нормалног стања. На основу добијених резултата може се закључити да иницијална имплементација омогућава поуздану детекцију аномалија над подацима прикупљеним са температурног сензора Raspberry Pi Pico W уређаја.

6.2 Евалуација система са наизменичним обучавањем и закључивањем

Након верификације исправности иницијалног система за детекцију аномалија, спроведена је евалуација система са наизменичним режимима обучавања и закључивања. За разлику од иницијалне имплементације, у којој се модел обучава једном и затим користи искључиво за процес закључивања, предложени систем омогућава периодично ажурирање током рада. Циљ ове евалуације био је да се испита да ли периодично ажурирање модела омогућава да систем и након сваког поновног обучавања настави поуздано да детектује аномалије над новоприкупљеним подацима са температурног сензора.

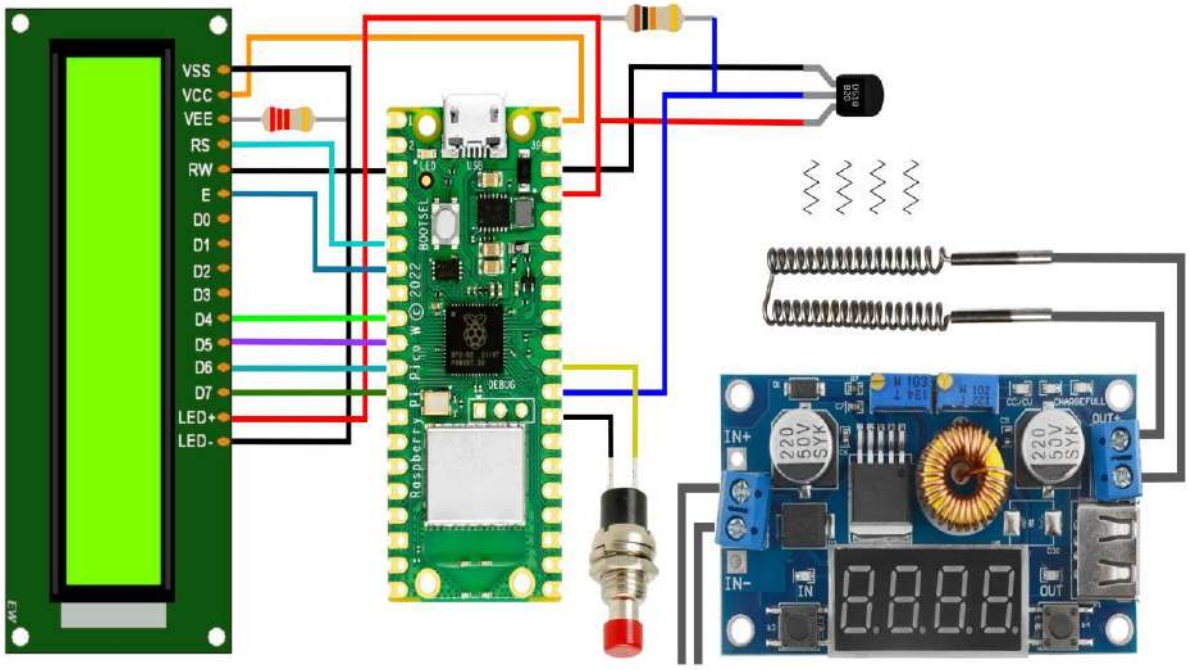
6.2.1 Експериментална поставка



Слика 10: Експериментална поставка

На слици 10 је приказана експериментална поставка са два Raspberry Pi Pico W уређаја, од којих је један температурни сензор изложен контролисаном загревању, док је сензор другог уређаја изложен собној температури. Централни рачунар представља серверски чвор и служи за покретање експеримената и праћење извршавања система.

За потребе евалуације система са наизменичним режимима обучавања и закључивања реализована је експериментална поставка која, поред компоненти коришћених у иницијалној имплементацији, обухвата и додатне хардверске елементе који омогућавају поузданију валидацију резултата и праћење понашања система током експеримента. Експериментална поставка обухвата све компоненте неопходне за реализацију процеса обучавања, закључивања и праћења рада система у реалном времену. Шематски приказ експерименталне поставке за контролисано загревање сензора приказан је на слици 11.



Слика 11: Шема експерименталне поставке

Експериментална поставка за контролисано загревање састоји се од Raspberry Pi Pico W уређаја, температурног сензора DS18B20, 16x2 LCD дисплеја, тастера, струјно контролисаног извора напајања и грејача који служе као извор топлоте. Додатне компоненте уведене су са циљем да омогуће контролисано изазивање промена температуре, праћење стања система током рада и поузданију валидацију резултата експерименталне евалуације. LCD дисплеј је коришћен за приказ тренутне вредности температуре очитане са сензора, чиме је омогућено континуирано праћење промене температуре током извођења експеримента. Тастер је додат како би се омогућило једноставно управљање експериментом и покретање одређених фаза рада система.

Конкретно он је додат како би се омогућило покретање мерења у тренутку када сензор достигне жељену температуру, на овај начин је олакшано понављање експеримената под приближно истим условима и обезбеђена боља упоредивост добијених резултата. Температурни сензор представља извор улазних података, на основу којих се врши закључивање и поновно обучавање модела. За разлику од иницијалне евалуације, у којој су промене температура изазиване ручно, у овој поставци је за загревање сензора коришћен контролисани извор топлоте, реализован помоћу грејача и струјно контролисаног извора напајања. Подешавањем струје грејача контролисано је загревање температурног сензора. На овај начин омогућено је контролисано и поновљиво изазивање промена температуре, што доприноси поузданијој валидацији резултата.

6.2.2 Почетна евалуација система

Као у претходним евалуацијама, почетна тачка анализе била је тачка E’, која одговара конфигурацији параметара (6, 15, 200). На основу ове конфигурације најпре је формиран иницијални модел, који је затим коришћен као полазни модел током евалуације система са наизменичним режимима обучавања и закључивања. За доношење одлука о присуству аномалије коришћен је исти оптимални праг одлучивања вредности 0.81, одређен као вредност за коју F1 мера достиже максимум на валидационом скупу података. Након формирања иницијалног модела започета је експериментална евалуација система, током које су се наизменично извршавали процеси закључивања и поновног обучавања. У наставку је приказан ток евалуације и резултати добијени у почетној имплементацији система.

Почетна евалуација система спроведена је кроз два понављања сличног експерименталног сценарија. Током сваког понављања бележени су редни број стабла које се ажурира, температуре очитане са сензора и резултат детекције аномалије. Циљ експеримента био је да се испита како се модел понаша када се током рада постепено излаже новим температурним вредностима и када се изолациона шума периодично ажурира на основу новоприкупљених података. У првој фази експеримента сензор је загреван до температуре од приближно 33 °C, а систем је ову вредност квалификовао као аномалију, што је било очекивано, јер је она одступала од података на којима је иницијални модел обучен. Након што је систем одређени број пута ажурирао модел на основу новоприкупљених температура, иста вредност више није класификована као аномалија, што указује на то да се модел прилагодио новом температурном опсегу. Након тога температура је повећана на приближно 36 °C, где систем поново у почетку детектује аномалије, али је након неколико циклуса ажурирања модела престао да класификује овај

температурни опсег као аномалан. Исти образац понашања уочен је и при повећању температуре на приближно 45 °C. Међутим, након додатног повећања температуре на приближно 55 °C, систем више није поуздано детектовао аномалије, даљим повећањем температуре није дошло до поновне детекције аномалија, већ је модел наставио да нове вредности класификује као нормално стање, што је указало на ограничење почетне имплементације система.

Слика 12: Промена температуре и понашања система током првог експеримента

Слика 13: Промена температуре и понашање система током другог експеримента

Слика 12 и слика 13 приказују резултате два понављања сличног експерименталног сценарија, при чему је приказано понашање система током постепеног повећавања температуре и узастопног ажурирања модела. Са слике 12 може се уочити да је систем сваки нови температурни опсег у почетку класификовао као аномалију. Међутим након

већег броја ажурирања модела систем је престао поуздано да детектује аномалије и при значајно вишим температурама.

На основу добијених резултата закључено је да механизам периодичног ажурирања модела омогућава прилагођавање новим температурним опсезима, али да у почетној имплементацији долази до постепеног губитка способности модела да разликује нормалне вредности од израженијих одступања. Ово ограничење представљало је основу за увођење додатног унапређења система, описаног у наредном одељку.

6.2.3 Унапређење система

Резултати почетне евалуације показали су да фиксна вредност прага аномалије, одређена током иницијалног обучавања модела, није обезбеђивала поуздану детекцију аномалија након вишеструких ажурирања изолационе шуме. Како се модел током рада постепено прилагођавао новоприкупљеним подацима, расподела вредности аномалијског скора се мењала, док је праг остајао непромењен. Као последица тога систем је временом губио способност да поуздано разликује нормална стања од аномалија. Због тога је у систем уведен механизам за динамичко одређивање прага аномалије. За разлику од почетне имплементације, у којој је праг имао фиксну вредност од 0.81 добијену током иницијалне евалуације модела, унапређена имплементација након сваког ажурирања изолационе шуме поново израчунава оптималну вредност прага. На тај начин праг се прилагођава тренутном стању модела и расподели аномалијског скора, чиме се обезбеђује поузданије доношење одлуке о присуству аномалије.

Прва измена у односу на почетну имплементацију односила се на начин одређивања прага аномалије. Иако је у претходним фазама развоја већ постојала функција за израчунавање оптималног прага, њено директно извршавање на Raspberry Pi Pico W уређају није било могуће због ограничења MicroPython и недостатка подршке за библиотеке које се користе у том поступку, из тог разлога израчунавање оптималног прага је премештено на серверску страну система. Функција која рачуна оптимални праг прима два параметра, један је аномалијски скор а други параметар су лабеле. За израчунавање аномалијског скора неопходан је обучени модел изолационе шуме. Из тог разлога је серверска имплементација проширена тако да, поред постојећих функционалности, формира и сопствену инстанцу модела изолационе шуме. На тај начин сервер располаже моделом који може да се користи за израчунавање вредности аномалног скора неопходног за израчунавање новог оптималног прага. Поред модела за израчунавање оптималног прага било је потребно и обезбедити и лабеле, пошто се оне добијају на основу синтетички генерисаног скупа података, било је потребно да клијенти серверу проследе

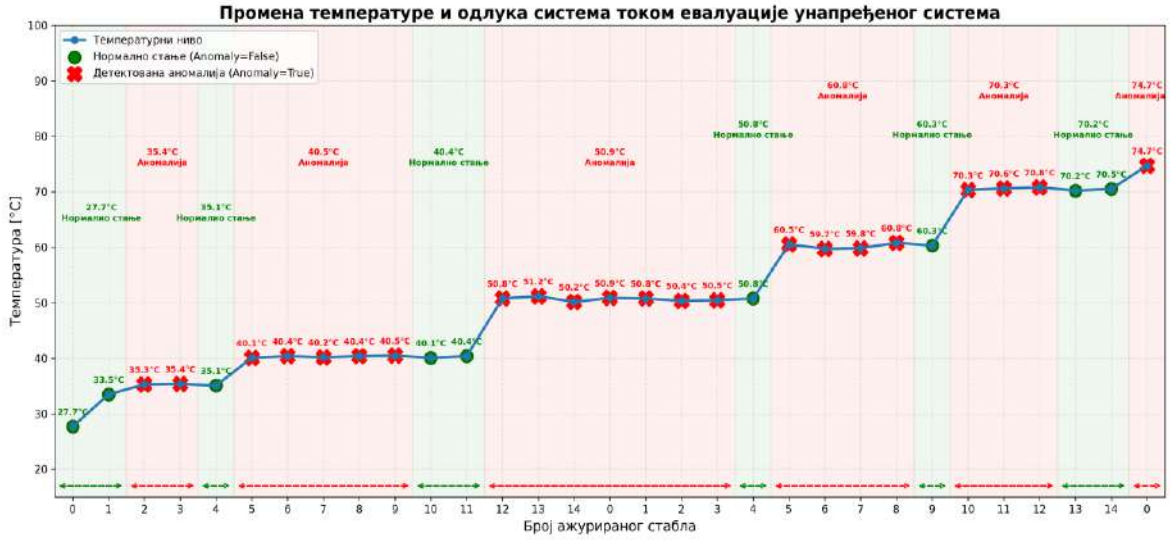
минималне и максималне вредности температура из својих локалних скупова података, како би сервер на основу примљених вредности одредио глобални минимум и максимум и затим их искористио за генерисање новог синтетичког скупа података. Над тако формираним скупом података израчунавају се вредности аномалијског скора и одређује се нова вредност оптималног прага, која се након тога враћа свим клијентима и користи у наредним циклусима закључивања.

Описане измене омогућиле су да оптимални праг аномалије након сваког ажурирања модела поново одређује на основу његовог тренутног стања, чиме је отклоњено ограничење фиксног прага примењеног у почетној имплементацији. У наставку је приказана евалуација унапређеног система, са циљем да се испита да ли уведене измене омогућавају очување поуздане детекције аномалија и након већег броја циклуса поновног обучавања.

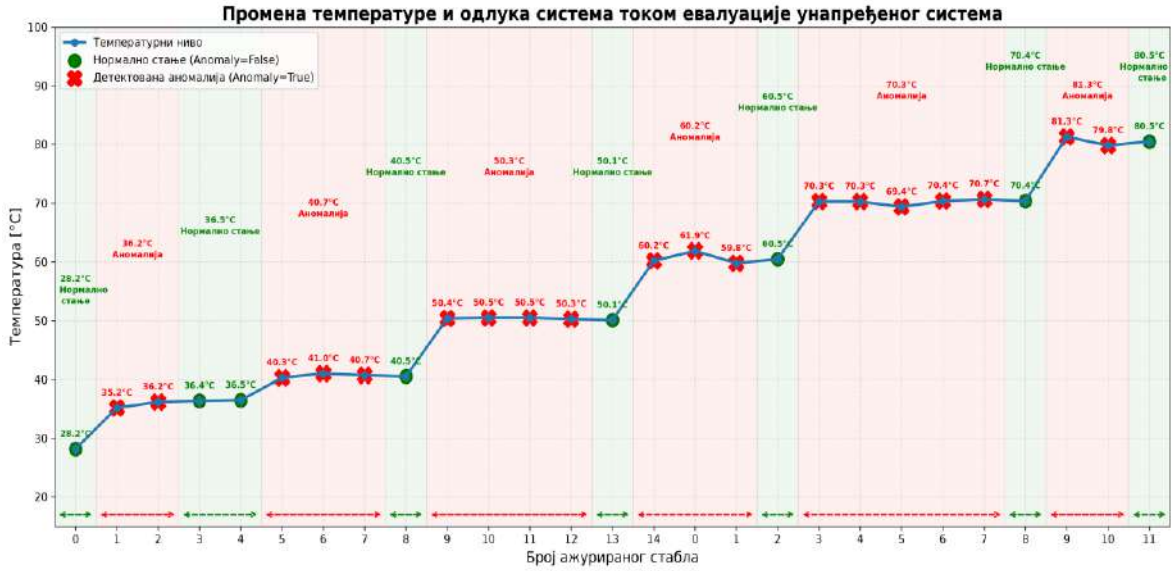
6.2.4 Евалуација унапређеног система

Након имплементације описаних измена извршена је поновна евалуација система са циљем да се испита њихов утицај на процес детекције аномалија током дуготрајног рада. Евалуација је спроведена под истим експерименталним условима као и код почетне имплементације, како би се омогућило директно поређење добијених резултата и објективна процена ефеката уведених унапређења. Експеримент је поново изведен кроз више понављања постепеним повећавањем температуре сензора, при чему је праћено понашање система током процеса ажурирања модела и поновног одређивања прага.

Резултати добијени током евалуације приказани су на сликама 14 и 15, које представљају два независна понављања истог експерименталног сценарија. На графицима је приказана промена температуре током експеримента, одлука система о присуству аномалије и ток адаптације модела након сваког ажурирања.



Слика 14: Понашање унапређеног система током првог експеримента



Слика 15: Понашање унапређеног система током другог експеримента

Са приказаних резултата може се уочити да систем и након више циклуса ажурираних модела наставља да успешно детектује сваки нови температурни опсег као аномалију. За разлику од почетне имплементације, након прилагођавања модела новим подацима не долази до губитка способности детекције аномалија при наредним повећањима температуре. То указује да динамичко одређивање прага омогућава очување очекиваног понашања система током читавог процеса рада система. Добијени резултати показују да је предложено унапређење успешно отклонило ограничење уочено током почетне евалуације. На тај начин омогућено је да се модел континуирано прилагођава новоприкупљеним подацима, без нарушавања способности система да благовремено препозна нове аномалије. Ово потврђује да увођење измена представља ефикасно решење за дуготрајан рад система у условима постепене промене температурних услова.

6.3 Претње за валидност

Током евалуације на Raspberry Pi Pico W уређају уочена је једна од кључних претњи за валидност, а то је нестабилност приликом мрежне комуникације. У случајевима када клијентски чворови нису у могућности да успоставе везу са серверским чвором, долази до прекида размене података, што за последицу има неуспешно извршавање алгоритма.

Нестабилност мрежне комуникације додатно отежава процес обучавања модела, посебно имајући у виду да извршавање једне итерације захтева више минута, што негативно утиче на укупно време извршавања и повећава вероватноћу прекида. Због тога су приказани резултати засновани искључиво на успешно извршеним експериментима спроведеним у условима стабилне мрежне повезаности, док су извршавања погођена нестабилношћу искључена из анализе, услед чега може бити нарушена поновљивост резултата у нестабилним мрежним условима.

Још једна претња валидности добијених резултата односи се на тачност мерења на различитим платформама. С обзиром на то да MicroPython на Raspberry Pi Pico W уређају и оперативни систем на PC платформи користе различите механизме управљања ресурсима и распоређивања задатака, може доћи до одступања у прецизности мерења времена извршавања.

7. Закључак

У овом раду представљен је систем за детекцију температурних аномалија, реализован адаптацијом PFLiForest федеративног алгоритма за детекцију аномалија намењен уграђеним уређајима као што су IoT уређаји. PFLiForest је алгоритам заснован првенствено на PTB-FLA окружењу, при чему је уследила његова адаптација за MPT-FLA окружење са циљем извршавања алгоритма на уређајима са ограниченим ресурсима. Поред адаптације алгоритма, реализован је и целокупан систем са наизменичним режимима обучавања и закључивања, као и експериментална поставка која је омогућила евалуацију система над реалним подацима. Такође је спроведена целокупна експериментална евалуација како би се процениле перформансе и ефикасност система у окружењима са ограниченим ресурсима.

Главне предности предложеног решења огледају се у реализацији целокупног система за детекцију температурних аномалија, који омогућава аутономан и континуиран рад кроз прикупљање података, детекцију аномалија у реалном времену и периодично поновно обучавање модела. Реализација оваквог система омогућена је адаптацијом PFLiForest алгоритма на MPT-FLA окружење, чиме је омогућено његово извршавање на Raspberry Pi Pico W уређају. При томе су постигнуте готово идентичне вредности AUC-ROC и AUC-PR метрика у односу на резултате приказане у полазном раду [2], чиме је потврђено да адаптација алгоритма није довела до нарушавања његових перформанси. Експериментална евалуација је показала да предложени систем успешно детектује температурне аномалије над реалним подацима прикупљеним са температурних сензора. Поред тога, увођењем система са наизменичним режимима обучавања и закључивања омогућено је континуирано прилагођавање модела новим подацима, док је имплементација динамичког одређивања прага за детекцију аномалија обезбедила стабилно понашање система.

Основно ограничење предложеног решења односи се на време обучавања модела на микроконтролерским платформама. Због ограничења процесорских и меморијских ресурса Raspberry Pi Pico W уређаја, поступак обучавања траје значајно дуже у односу на извршавање истог алгоритма на персоналним рачунарима. Такође, приликом тумачења добијених резултате експерименталне евалуације потребно је узети у обзир претње за валидност разматране у одељку 6.3, које могу утицати на поновљивост и прецизност добијених резултата.

Овај рад представља основу за даља истраживања у више праваца. Први правац односи се на спровођење обимније експерименталне евалуације, у оквиру које би се систематски испитивао утицај различитих вредности параметара предложеног решења на квалитет добијених резултата, уместо њиховог задржавања на фиксним вредностима. Други правац односи се на даље унапређење механизма за динамичко одређивање прага аномалије, кроз испитивање алтернативних приступа који би могли да обезбеде још боље прилагођавање различитим условима рада. Посебно интересантан правац будућих истраживања односи се на прецизније дефинисање самог појма аномалије у системима који се континуирано прилагођавају новим подацима. Иако предложени систем успешно усваја нове температурне опсеге као нормално стање, поставља се питање како дефинисати апсолутне аномалије, као што су екстремно високе температуре које могу указивати на појаву пожара, а које би требало препознати као аномалије без обзира на претходну адаптацију модела. Истраживања механизама које би истовремено омогућили адаптацију модела и задржали способност детекције критичних догађаја представља значајан правац будућег рада.

8. Литература

[1] A. Zunic, P. Vasiljevic and M. Popovic, " Anomaly detection system at the edge of the internet based on the MPT-FLA," in IEEE Zooming Innovation in Consumer Technologies Conference (ZINC), Novi Sad, Serbia, 2026.

[2] P. Vasiljevic, M. Matic and M. Popovic, "Federated Isolation Forest for Efficient Anomaly Detection on Edge IoT Systems," in IEEE Zooming Innovation in Consumer Technologies Conference (ZINC), Novi Sad, Serbia, 2025. doi: 10.1109/zinc65316.2025.11103552.

[3] M. Popovic, M. Popovic, I. Kastelan , M. Djukic и S. Ghilezan, „A Simple Python Testbed for Federated Learning Algorithms,“ у 2023 Zooming Innovation in Consumer Technologies Conference (ZINC), Novi Sad, Serbia, 2023. doi: 10.1109/zinc58345.2023.10173859.

[4] M. Popovic, M. Popovic, I. Kastelan, M. Djukic и I. Basicevic, „MicroPython Testbed for Federated Learning Algorithms,“ у 2024 32nd Telecommunications Forum (TELFOR), 2024. doi: 10.1109/telfor63250.2024.10819071.

[5] M. Popovic, „GitHub repository PTB-FLA,“ 2025. [На мрежи]. Available: https://github.com/miroslav-popovic/ptbfla. [Последњи приступ Oct. 2025].

[6] V. Chandola, A. Banerjee и V. Kumar, „Anomaly detection: A Survey,“ т. 41, бр. 3, pp. 1-58, Jul. 2009. doi: 10.1145/1541880.1541882.

[7] P. Kairouz, H. B. McMahan, B. Avent и A. Bellet, „Advances and Open Problems in Federated Learning,“ 2019. doi: 10.1561/2200000083.

[8] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated Machine Learning: Concept and Applications,” arXiv, 2019, doi: 10.48550/ARXIV.1902.04885.

[9] H. Xiang, X. Zhang, X. Xu, A. Beheshti, L. Qi, Y. Hong and W. Dou, "Federated Learning-Based Anomaly Detection with IsolationForest in the IoT-Edge Continuum," in *ACM Transactions on Multimedia Computing, Communications, and Applications*, , vol. 22, no. 1, pp. 1–19, Jan. 2026, doi: 10.1145/3702995

[10] J. Li, "Federated anomaly detection with Isolation Forest in the IoT network," Macquarie University, 2024, doi: 10.25949/25286269.V1.

[11] H. Ochiai, R. Nishihata, E. Tomiyama, Y. Sun, and H. Esaki, "Detection of Global Anomalies on Distributed IoT Edges with Device-to-Device Communication," Proceedings of the Twenty-fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing. ACM, pp. 388–393, Oct. 16, 2023. doi: 10.1145/3565287.3616528.

[12] M. Gulati, K. Zandberg, Z. Huang, G. Wunder, C. Adjih, and E. Baccelli, "TDMiL: Tiny Distributed Machine Learning for Microcontroller-Based Interconnected Devices," IEEE Access, vol. 12, pp. 167810–167826, 2024, doi: 10.1109/access.2024.3492921.

[13] D. Pau, A. Khiari, and D. Denaro, "Online learning on tiny micro-controllers for anomaly detection in water distribution systems," 2021 IEEE 11th International Conference on Consumer Electronics (ICCE-Berlin). IEEE, pp. 1–6, Nov. 15, 2021. doi: 10.1109/icce-berlin53567.2021.9720009.

[14] M. Antonini, M. Pincheira, M. Vecchio, and F. Antonelli, "An Adaptable and Unsupervised TinyML Anomaly Detection System for Extreme Industrial Environments," Sensors, vol. 23, no. 4, p. 2344, Feb. 2023, doi: 10.3390/s23042344.

[15] C. Feng, N. Huber, A. H. Celdrán, G. Bovet, and B. Stiller, "Demo: A Practical Testbed for Decentralized Federated Learning on Physical Edge Devices," 2025 IEEE 50th Conference on Local Computer Networks (LCN). IEEE, pp. 1–3, Oct. 13, 2025. doi: 10.1109/lcn65610.2025.11146362.