БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ

ИНСТИТУТ ПО ИНФОРМАЦИОННИ И КОМУНИКАЦИОННИ ТЕХНОЛОГИИ

СЕКЦИЯ “ИНФОРМАЦИОННИ ПРОЦЕСИ И СИСТЕМИ ЗА ВЗЕМАНЕ НА РЕШЕНИЯ”

Илиян Магдаленов Барзев

ИЗСЛЕДВАНЕ И АНАЛИЗ НА ВЪЗМОЖНОСТИТЕ ЗА ОТКРИВАНЕ НА ЗЛОНАМЕРЕН СОФТУЕР ЧРЕЗ СРЕДСТВАТА НА МАШИННО ОБУЧЕНИЕ

АВТОРЕФЕРАТ

на дисертация

за присъждане на образователна и научна степен “Доктор”

Професионално направление: 4.6. “Информатика и компютърни науки”

Докторска програма “Информатика”

Научен ръководител

проф. д.н. Даниела Борисова

2026

Дисертационният труд е обсъден и допуснат до защита на разширено заседание на секция „Информационни процеси и системи за вземане на решения“ при ИИКТ-БАН, състояло се на 15.12.2025.

Дисертационният труд е структуриран в увод, 3 глави, заключение, приноси, списък на публикации, декларация за оригиналност на резултатите и библиография. Дисертационният труд е в общ обем от 143 страници, 20 таблици, 43 фигури и 155 литературни източника.

Защитата на дисертацията ще се състои на ……..2026 г.
от …… часа в зала …….. на блок 2 на ИИКТ-БАН
на открито заседание на научно жури в състав:

Научно жури
1. ……
2. ……
3. ……
4. ……
5. ……

Рецензиите и становищата на членовете на научното жури и авторефератът са публикувани на сайта на ИИКТ-БАН.

Материалите за защитата са на разположение на интересуващите се в стая 315 на ИИКТ-БАН, ул. „Акад. Г. Бончев“, бл. 2.

Автор: **Илиян Магдаленов Барзев**

Заглавие: **Изследване и анализ на възможностите за откриване на злонамерен софтуер чрез средствата на машинно обучение**

УВОД

Днес защитата от зловреден софтуер е важна по няколко причини. Едната се отнася до сигурността на данните, тъй като злонамереният софтуер може да компрометира лични и финансови данни, което води до кражба на самоличност или финансови загуби. Втората е свързана със състоянието на системите, тъй като инфекциите с вируси и троянски коне могат да повредят компютрите и мрежите, което води до загуба на информация и скъпи ремонти. Злонамереният софтуер може също да повлияе на производителността, като причинява сривове, нарушаващи ежедневните операции и загуба на време и ресурси. Всички тези ситуации се отразяват на репутацията на компаниите – компрометирането на данни може да повлияе на доверието на клиентите, което е трудно за възстановяване. Не на последно място е спазването на законите и разпоредбите, тъй като много организации са задължени по закон да защитават данните на своите клиенти. Нарушенията могат да доведат до сериозни санкции. Злонамереният софтуер може да се разпространява от едно устройство на друго, застрашавайки цялата мрежова сигурност.

Анализът на зловреден софтуер е процесът на локализиране и изследване на зловреден софтуер или код с цел разбиране на неговото действие и разработване на контрамерки. Зловреден софтуер може да приеме много форми, като вируси, червеи, троянски коне и ransomware, и може да причини значителни вреди на хора, организации и дори цели държави. За да се определи целта, потенциалните ефекти и възможности на даден зловреден софтуер, анализът на зловреден софтуер включва изследване на поведението, структурата и функционалностите на зловредния софтуер. Анализаторите на зловреден софтуер са от съществено значение за сектора на киберсигурността, защото се стремят да откриват опасности, да ги елиминират и да се защитават от онлайн атаки. Чрез използване на знанията, получени от анализа на зловреден софтуер, могат да се създадат решения за сигурност, които ще защитят по-добре бизнеса от опасен софтуер. Анализът на зловреден софтуер е ключова част от всяка успешна стратегия за киберсигурност в непрекъснато променящия се пейзаж на заплахите днес.

С еволюцията на дигиталните технологии се променят и методите за извършване на атаки, което налага разработване на нови алгоритми за ефективно и навременно откриване на зловредно поведение. Традиционните решения, базирани предимно на сигнатури, вече не са достатъчни за справяне със съвременните заплахи. Исторически антивирусните системи използват сигнатурно-базирани алгоритми, които сравняват файлове с база данни от вече известни вредители. Този подход има няколко ключови недостатъка: Не идентифицира нова или мутирала заплаха; Изисква непрекъснато обновяване на бази данни; Неефективен срещу полиморфен, метаморфен и fileless malware. Развитието на зловредния софтуер значително изпреварва възможностите на сигнатурните системи. От друга страна, днешният malware използва усъвършенствани техники за укриване и адаптация – кодът се променя при всяко изпълнение; а атаките се

изпълняват само в паметта, автоматично се генерират и нови варианти. Всички тези фактори повишават сложността на защитата и изискват нови методи за откриване.

Ръстът на атаките експоненциално нараства, като по данни на световни изследователски центрове ежедневно се появяват десетки хиляди нови модификации. Налице е неотложна необходимост от проактивна защита. Съвременните системи трябва да откриват заплахи преди да са вредили, а това би могло да се реализара чрез машинно обучение и поведенчески анализ, който изследване на поведението, структурата и функционалностите на malware.

Следователно, разработването на нови алгоритми за откриване на зловреден софтуер е критично за осигуряване на информационна сигурност в съвременната дигитална среда. Еволюцията на заплахите налага използването на интелигентни, адаптивни и мащабируеми подходи, които могат да реагират на динамиката на модерните кибератаки. Инвестициите в иновации и алгоритмични модели са ключови за изграждане на устойчиви защитни системи.

Като се взимат предвид всички тези аспекти, става ясно, че провеждането на изследвания и анализи на възможностите за откриване на злонамерен софтуер чрез средствата на машинно обучение е една актуална и бързо развиваща се научна област.

Изложението на дисертационния труд е структурирано в увод, три глави, заключение – резюме на получените резултати, приноси, списък на публикациите, списък на забелязаните цитирания, декларация за оригиналност и библиография.

В **Глава 1** е направен анализ на различни алгоритми на машинното обучение относно тяхното представянето за целите на откриване на зловреден софтуер. Представено е тестване на приложимостта на алгоритми за двоична класификация за откриване на зловреден софтуер, използвайки публичен набор от данни, заразен с 9 вида зловреден софтуер, чрез предложена методология. Следвайки тази методология и определените показатели, резултатите показват приложимост на изследвани алгоритми, като един от тях (Random Forest) демонстрира по-добра производителност. Показано е, че хибридният модел CNN-LSTM значително превъзхожда други тесвани подходи, постигайки точност от 0,97 и също толкова високи стойности за прецизност, попълнота и F1-Score. Този хибриден модел комбинира силните страни на възможностите за извличане на характеристики на CNN със способността на LSTM да улавя времеви зависимости, което го прави особено ефективен за данни от IoT-23.

В **Глава 2** е описан предложен модел за избор на виртуална машина за целите на провеждането на експерименти за откриване на зловреден софтуер. Представен е подобрен подход на статичен анализ чрез оптимизиране извличането на характеристики, комбинирайки различни алгоритми за машинно обучение за откриване на зловреден софтуер. Представена е предложена рамка за статична класификация на зловреден софтуер, използваща оптимизация на функции и ансамблово обучение. За подобряване на класификация на зловреден софтуер е предложена самоосъзната рамка, интегрираща

рутиране на модели на базата на система за доверие за избора и обяснимост на характеристиките. За прецезиране на класификация на зловредния софтуер е предложена адаптивна рамка, съобразена с доверието, позволяваща класификация на зловреден софтуер с възможност за корекции чрез обратна връзка. Тази адаптивна рамка е реализирана в разработена демо версия на софтуерно приложение под името „Shipka Guard“. Тя интегрира механизъм за рутиране, съобразен с доверието, който използва множество поколения модели, включително резервен механизъм. В нея се използва буфер за обратна връзка и слой за корекции за лека адаптация от потребителски корекции и синтетични инжекции.

В **Глава 3** са представени резултатите от проведеното тестване на предложените модели за избор на софтуер за виртуална машина. Описани са резултати от теставане на предложения подобрен подход на статичен анализ чрез оптимизиране извличането на характеристики, комбинирайки различни алгоритми за машинно обучение. Описани са числени експерименти, използвайки предложената рамка за статична класификация на зловреден софтуер, в която е направено оптимизиране на функциите и е използавно ансамблово обучение. Показани са също резултати от теставаната самоосъзната рамка, интегрираща рутиране на модели на базата на система за доверие за избора и обяснимост на характеристиките. Тук е преставено разработеното и тествано приложение Shipka Guard, интегриращо предложената адаптивна рамка, съобразена с доверието, позволяваща класификация на зловреден софтуер с възможност за корекции чрез обратна връзка. Показано е, че буферът за обратна връзка дава възможност на потребителя съвместно да коригира модела, което му позволява да експортира/импортира файлове с обратна връзка. От друга страна, интеграцията на обяснимостта допринася за доверието в решенията. Всичко това е реализирано в разработена демо версия на софтуерно приложение под името „Shipka Guard“.

В заключението е направено обощение на получените резултати в следствие на проведените изследвания, предмет на настоящия дисертационен труд. Показани са някои основни посоки, който да се използват в бъдещи изследвания.

ГЛАВА 1. АНАЛИЗ НА ТЕХНИКИТЕ ЗА ОТКРИВАНЕ И КЛАСИФИЦИРАНЕ НА ЗЛОВРЕДЕН СОФТУЕР

В настоящата глава е направен анализ и сравнение на различни алгоритми на машинното обучение относно тяхното представяне за целите на откриването на зловреден софтуер.

1.1. Анализ и класификация на алгоритмите на машинно обучение за откриване на злонамерен софтуер

Машинното обучение (МО) като подобласт на изкуствения интелект предоставя възможност за правене на прогнози въз основа на модели, научени директно от данни,

без да са изрично програмирани за това (Ozkan-Okay et al., 2024). МО заема ключова роля в откриването и смекчаване на сложните заплахи на зловредния софтуер по различни начини. Чрез средствата на МО могат да анализират големи обеми от данни, да се класифицират файлове и програми, да се прогнозират заплахи и др. (Фиг. 1.1).

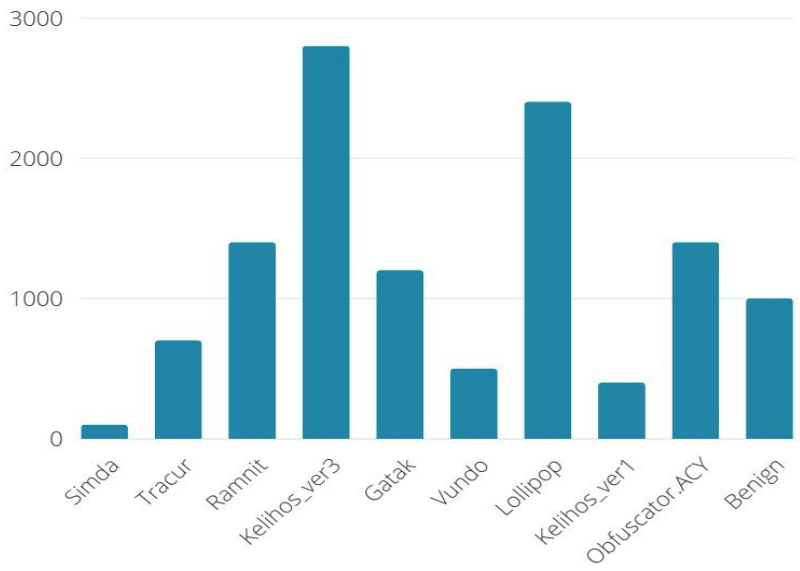
Фиг. 1.1. Приложение на машинното обучение

1.2. Сравнение на различни алгоритми за двоична класификация

За целите на сравнението на приложимостта и точността на различни алгоритми за откриване на зловреден софтуер в един и същ обем данни се използва следвана следната диаграма, както е показано на Фиг. 1.2 (Barzev et al., 2024).

Фиг. 1.2. Блок-схема на методологията на експеримента

Наборът от данни, използван в този експеримент е BIG2015 Dataset. Той съдържа етикетирани доброкачествени и злонамерени PE и OLE файлове. Разпределението на зловредния софтуер в набора от данни е показано на Фиг. 1.3.



Фиг. 1.3. Представяне на разпространението на зловреден софтуер в набора от данни

Преди да се стигне до резултата от експеримента, е важно да се определят показателите, върху които ще се съсредоточим, за да се определи колко точно алгоритмите се представят спрямо набора от данни. Избраните показатели са: 1) Accuracy, 2) Precision), 3) Recall и 4) F-1 резултат.

Получените стойности на изследваните метрики за избраните алгоритми Random Forest, K-Nearest Neighbors и Support Vector Machines върху изследвания набор от данни са показани в Таблица 1.2.

Таблица 1.2. Резултати от използваните показатели

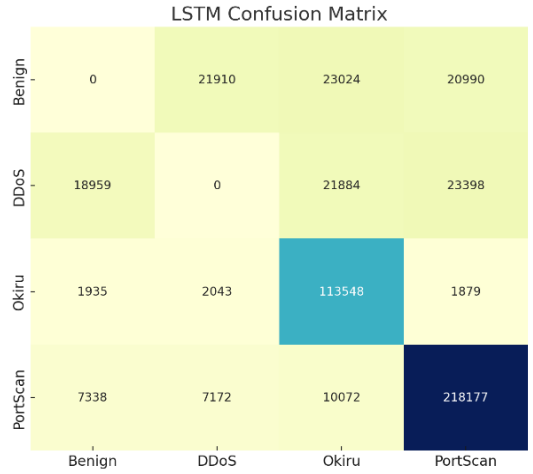
Алгоритъм	Метрики			
	Accuracy	**Precision**	**Recall**	**F-1**
RF (Random Forest)	0.9825	0.9812	0.9999	0.9893
K-NN (K-Nearest Neighbors)	0.8632	0.8911	0.9938	0.9731
SVM (Support Vector Machines)	0.7819	0.9653	0.7309	0.8487

От Таблица 1.2 може да се установи, че най-добрият алгоритъм за използваната метрика е Random Forest. Това се дължи на точността на Random Forest, която е най-добра със стойност 0.9825, следвана от K-NN със стойност 0.8632 и последна е Support Vector Machines с 0.7819. Относно вторият показател – прецизност, резултатите са аналогични и най-добра производителност показва алгоритъмът Random Forest, също със стойност 0.9812, следван от Support Vector Machines и накрая от K-NN. Това се дължи на получените резултати за избраните метрики. За останалите метрики, най-добри резултати се получават от алгоритъма Random Forest (Таблица 1.2).

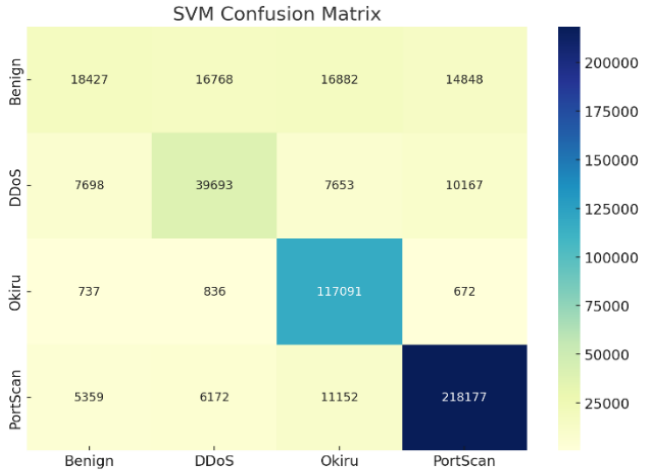
1.3. Анализ на производителността на алгоритми за откриване на зловреден софтуер в дейта сет от областта на интернет на нещата

Едно от критичните предизвикателства, свързани с проникването на зловреден софтуер, е получаването на неоторизиран достъп до IoT устройства, където зловредният софтуер се опитва да копира оторизирани устройства, като имитира техните хардуерни и софтуерни спецификации (Praveen et al., 2023).

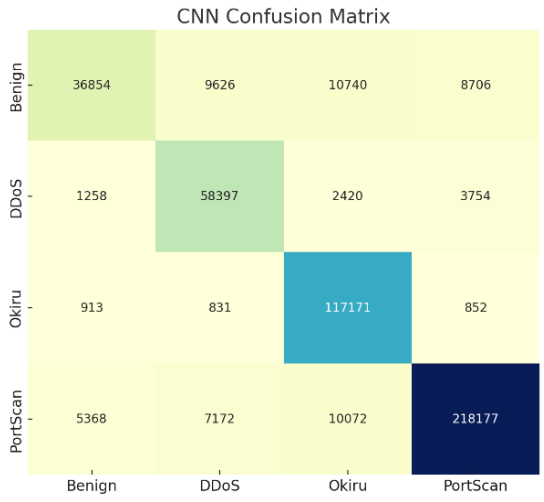
За анализа на производителността на алгоритмите LSTM, SVM, CNN и CNN-LSTM за откриване на зловреден софтуер са използвани данни за мрежовия трафик от извадка от набора от данни IoT-23. Съответните матрици на объркване, визуализиращи и обобщаващи производителността на алгоритмите за класификация, са както следва Фиг. 1.4 до Фиг. 1.7.



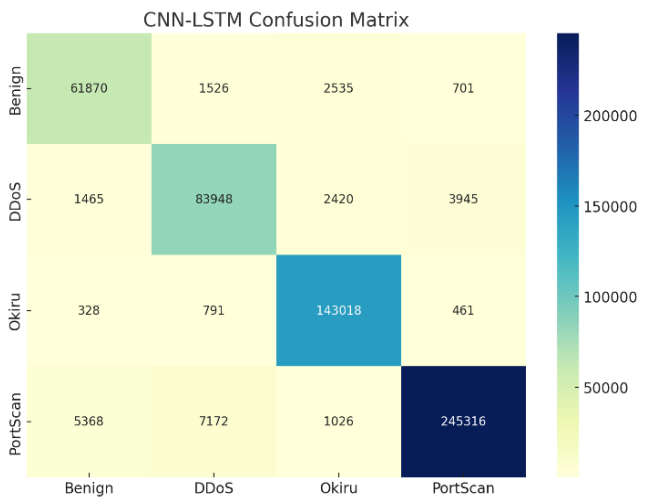
Фиг. 1.4. Матрица на объркване на LSTM



Фиг. 1.5. Матрица на объркване на SVM



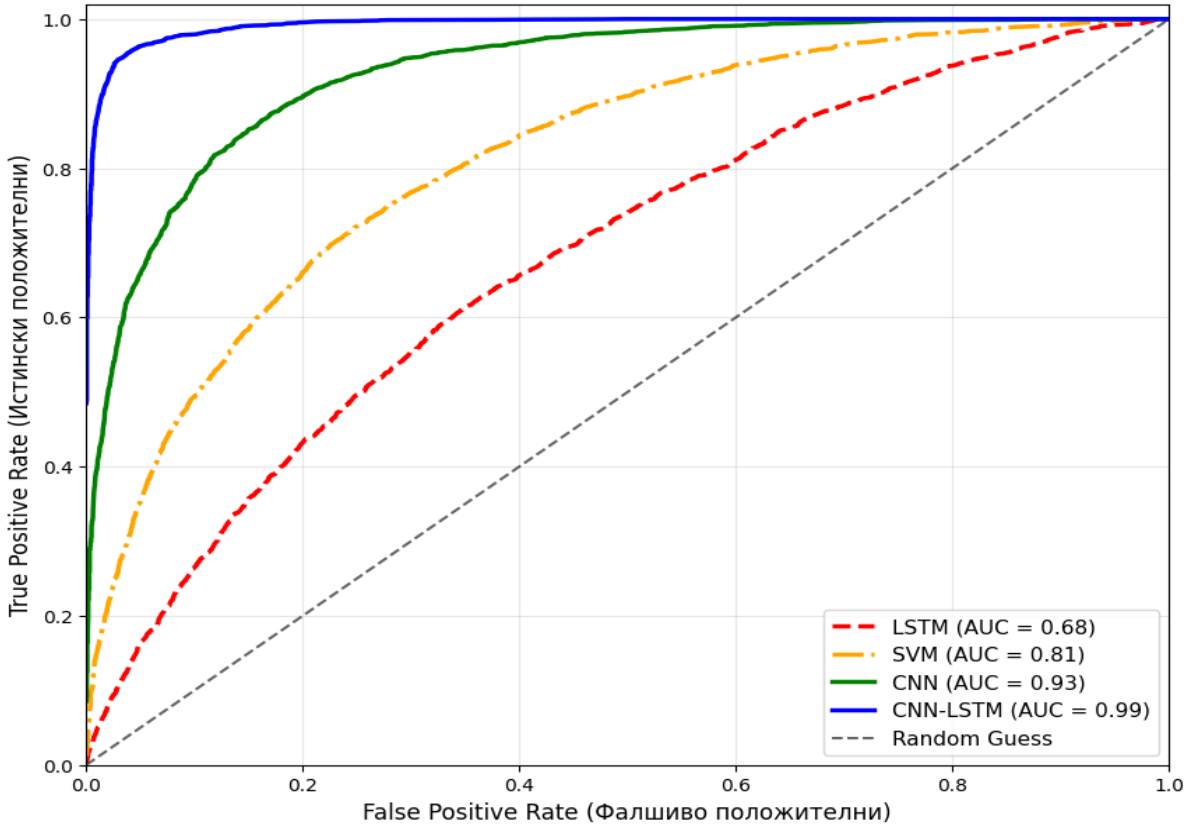
Фиг. 1.6. Матрица на объркване на CNN



Фиг. 1.7. Матрица на объркване на CNN-LSTM

За да се оцени производителността на разгледаните по-горе четири алгоритъма за класификация LSTM, SVM, CNN и CNN-LSTM, е използван анализ на кривата на оперативните характеристики на приемника (Receiver Operating Characteristic – ROC). ROC кривите графично илюстрират дискриминационните способности на всеки модел чрез

отразяване на истински положителния процент (TPR) спрямо фалшиво положителния процент (FPR) за различни прагови стойности, както е показано на Фиг. 1.8.



Фиг. 1.8. ROC криви за LSTM, SVM, CNN, CNN-LSTM

Чрез сравняването на ROC кривите на всеки модел може да се оценят компромисите между чувствителност и специфичност. Този анализ дава представа за способността на всеки алгоритъм правилно да идентифицира положителни случаи, като същевременно минимизира фалшиво положителните резултати, предлагайки цялостен поглед върху ефективността на класификацията при различни прагове. По-високата крива с по-голяма AUC предполага по-добра производителност при класифицирането на случаи. Формата и разположението на всяка крива отразяват компромисите между правилното идентифициране на положителни случаи и избягването на фалшиви положителни резултати.

За да се оцени цялостно класификационната ефективност на четирите модела – LSTM, SVM, CNN и CNN-LSTM – се анализираха няколко ключови показателя, включително точност, прецизност, попълнота на повторение и F1 резултат – Таблица 1.3.

Таблица 1.3. Резултати от производителността на алгоритмите

Модел	Accuracy	Precision	Recall	F1-Score
LSTM	0.74	0.72	0.74	0.71
SVM	0.84	0.86	0.84	0.84
CNN	0.91	0.92	0.91	0.91
CNN-LSTM	0.97	0.97	0.97	0.97

1.3. Изводи

Машинното обучение като област от изкуствения интелект, използва алгоритми, за да се учи от данни, да идентифицира модели и да прави прогнози или решения с минимална човешка намеса. Следователно, могат да се направят някои конкретни изводи от проведените анализи и сравнения, а именно.

- Съществува разнообразие от модели, базирани на машинно обучение, които са способни за откриване на злонамерен софтуер;
- Важен етап от прилагането на машинно обучение и по-специално на контролираното обучение е процесът на самото обучение, при което се цели обучаване с данни за конкретна предметна област;
- Комбинирането на различни модели на машинното обучение може да допринесе за преодоляване недостатъците на самостоятелно използваните модели.

Следователно, изброените по-горе предпоставки са основа за разработване на подобрени модели, рамки и приложения, водещи до по-добри резултати при откриването и класифицирането на зловреден софтуер.

1.4. Цел и задачи

Цел на дисертационният труд е да се изследват и анализират възможностите за откриване на злонамерен софтуер чрез средствата на машинно обучение, на база на които да се предложат подходящи хибридни модели, рамки и приложения за получаване на по-добри резултати при откриването на зловреден софтуер. За постигането на тази цел е необходимо да се изпълнят следните задачи:

1) да се направи анализ на различни алгоритми на машинното обучение относно тяхното представянето за целите на откриване на зловреден софтуер;
2) да се определи подходяща виртулна машина, която да се използва при провеждане на тестове за откриване и класификация на зловреден софтуер;
3) да се предложи подобрен подход за статичен анализ за откриване на зловреден софтуер чрез оптимизиране извличането на характеристики и комбинирайки различни алгоритми за машинно обучение;
4) да се предложи рамка за статична класификация на зловреден софтуер, използваща оптимизация на функции и ансамблово обучение;
5) да се предложи самоосъзната класификация на зловреден софтуер чрез рутиране на модели на базата на система за доверие за избора и обяснимост на характеристиките;
6) да се предложи адаптивна рамка, съобразена с доверието, за класификация на зловреден софтуер с корекции за обратна връзка.

ГЛАВА 2. МОДЕЛИ ЗА ИЗБОР НА ВИРТУАЛНА МАШИНА, ХИБРИДНИ АЛГОРИТМИ И РАМКИ ЗА ОТКРИВАНЕ НА ЗЛОНАМЕРЕН СОФТУЕР ЧРЕЗ МЕТОДИТЕ НА МАШИННОТО ОБУЧЕНИЕ

2.1. Избор на софтуер за виртуални машини за целите на откриване на зловреден софтуер

Като се има предвид важността на виртуалната машина не само за бизнес цели, но и за откриване на зловреден софтуер, са предложени два модела за групово вземане на решения за избор на виртуална машина.

2.1.1 Модели за групово вземане на решение при избор на софтуер за виртуална машина

Разгледан е конкретен проблем, отнасящ се до избор на виртуална машина, която да работи на десктоп с Windows. Като достатъчно общи показатели могат да се използват критериите „лекота на използване“, „обслужване на клиенти“ и „съотношение цена-качество“, които могат лесно да бъдат разбрани. Всички тези критерии, заедно с вече дадените оценки, се използват в следния предложен модел за групово вземане на решения (Borissova et al., 2023):

$$\max\{A_j^*\} = \sum_{q=1}^{Q} \lambda^q \sum_{i=1}^{I} w_i \, e_i^q \tag{2.1}$$

$$\sum_{i=1}^{I} w_i = 1 \tag{2.2}$$

$$\sum_{q=1}^{Q} \lambda^q = 1 \tag{2.3}$$

където индекс j е използван за обозначаване на набор от дадените алтернативи ($j = 1, \dots, J$), с индекс i е означено множеството на критериите за оценка ($i = 1, \dots, I$), а индекс q е използван за означаване на множеството на лицата, вземащи решение ($q = 1, \dots, Q$) които ще формират крайното групово решение. Коефициентите w_i изразявят важността на критериите за оценка и трябва да спазват отношението (2.2). Другият вид коефициенти e_i^q изразяват оценките на *i-ия* критерий съгласно гледната точка на *q-ия* експерт. В зависимост от компетентността на всеки експерт за всеки се задават съответни тегла λ^q, които ще участват във формирането на крайното групово решение.

Възможно е също така, вместо избор на един единствен тип софтуер за виртуална машина, даденият набор от алтернативи да се редуцира до подмножество, а изборът да се извърши по други критерии. Например, може да се направи последващо класиране, използвайки специфичните характеристики. В тази ситуация може да се използва следният комбинаторен оптимизационен модел (Borissova et al., 2023):

$$max\left\{\sum_{j=1}^{J} x_j \sum_{q=1}^{Q} \lambda^q \sum_{i=1}^{I} w_i \, e_i^q\right\} \tag{2.4}$$

При ограничения

$$\sum_{j=1}^{J} x_j = C, x_j \in \{0,1\} \quad (2.5)$$

$$x_j \in \{0,1\} - binary\ integers \quad (2.6)$$

$$C < J \quad (2.7)$$

$$\sum_{i=1}^{I} w_i = 1 \quad (2.8)$$

$$\sum_{q=1}^{Q} \lambda^q = 1 \quad (2.9)$$

Използваните двоични променливи x_j са свързани с всяка алтернатива, а константата C изразява броя на алтернативите, до които се стремим да редуцираме даденото множество алтернативи. Очевидно е, че тази константа трябва да е по-малка от броя на алтернативите, изразени с (2.7). Целевата функция (2.4) ще избере точно тези алтернативи с по-добри резултати, съобразно оценките e_i^q.

2.2. Подобрен подход на статичен анализ чрез оптимизиране извличането на характеристики, комбинирайки различни алгоритми за машинно обучение за откриване на зловреден софтуер

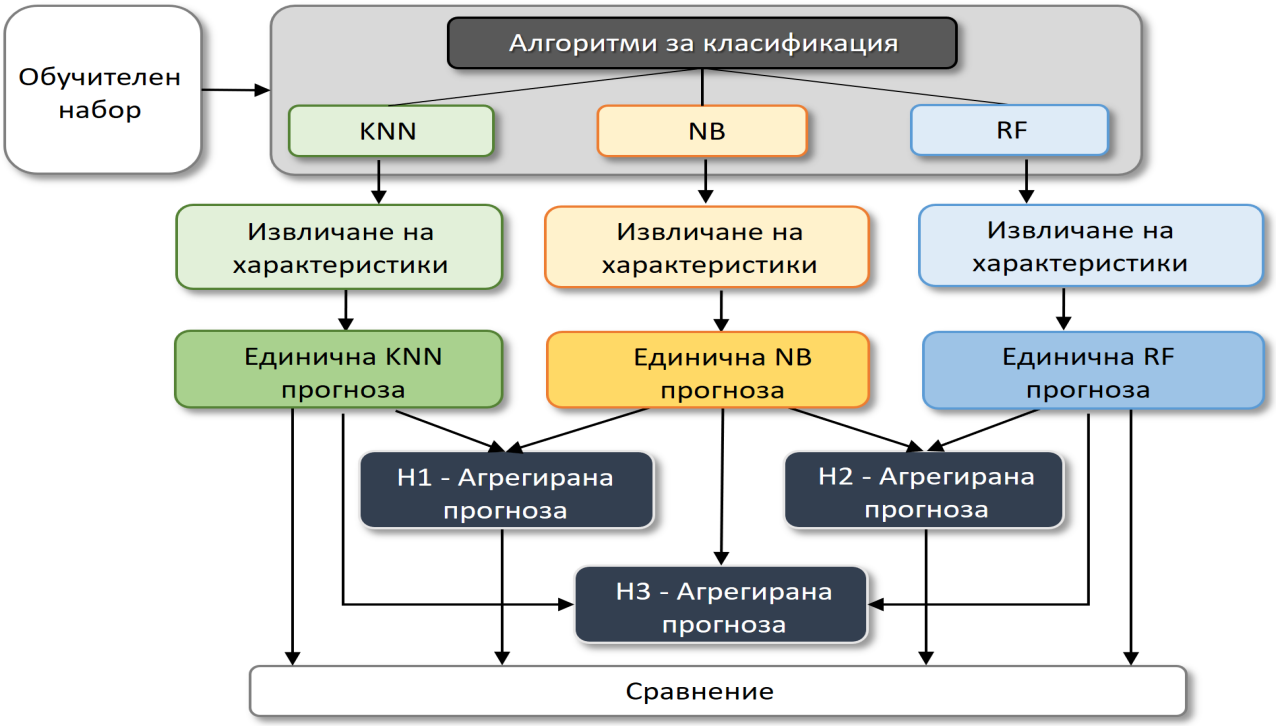
Статичният анализ на зловреден софтуер, като проактивен подход, включва анализ на характеристиките и компонентите на подозрителен файл, без той да се изпълнява. Този метод предоставя информация за структурата, поведението и потенциалните рискове на файла, като анализира различни атрибути, като заглавка на файла, метаданни, низове, ресурси и код. Този безопасен статичен подход се отличава с бързо сканиране и маркиране на често срещани заплахи, което го прави идеален за антивирусен софтуер и първоначално сортиране.

2.2.1. Подобрен подход за откриване на зловреден софтуер чрез комбиниране на различни алгоритми за машинно обучение

Обобщеното алгоритмично представяне на предложения подход за подобрено откриване на зловреден софтуер чрез комбиниране на различни алгоритми за машинно обучение в 3 хибридни алгоритъма е илюстрирано на Фиг. 2.2 (Barzev & Borissova, 2025).

В началната фаза се сравняват индивидуалните точности на класификаторите, а в следващата фаза – точността на формираните хибридни алгоритми:

1) H1 като комбинация на NB и KNN;
2) H2 като комбинация на NB и RF;
3) H3 като комбинация на NB, KNN и RF.



Фиг. 2.2. Обобщено алгоритмично представяне на предложения подход

NB като директен вероятностен класификатор работи при предположението за силна независимост на признаците. Той разчита на теоремата на Бейс, за да изчисли априорната вероятност $P(H)$ и апостериорната вероятност $P(H|E)$ за събитие, наблюдавано съответно преди и след доказателствата, и се изразява като правило на Бейс:

$$P\ (H|E)\ =\ (P(E|H)\ \mathrm{x}\ P(H))/P(E)$$

Псевдокодът за наивния алгоритъм на Бейс е следният:

```
Step 1: Training:
      For each feature
         For each feature-value E
             For each class-label H
             P(E/H) = total number of occurrences of feature value with class
label)/(total number of occurrences of class label)
             End for
         End for
      End for
Step 2: Testing:
      For each instance in test data
         Calculate probability using P(H/E)=(P(E/H)*P(H))/P(E)
      End for
Step 3: Assign the class label with the maximum probability to the test instance.
```

По подобен начин могат да бъдат изразени алгоритмите KNN и RF

След процеса на избор на характеристики, броят на определените такива в двата набора от данни е съответно тринадесет и петнадесет. Крайният резултат съдържа списък със 7 важни характеристики, както е показано в Таблица 2.1.

Таблица 2.1. Характеристики, използвани от XGB модела, с тяхното описание

No	Feature	Description
1	conn_state_50	Connection state
2	proto_tcp	Transaction protocol
3	idresp_p	Destination port
4	idresp_h	Destination IP address
5	resp_pkts	Destination packets
6	orig_ip_bytes	Source2destination transaction bytes
7	Idorig_p	Source port

Важно условие за постигане на по-добра точност при комбиниране на класификатори е използването на различни набори от характеристики или различни обучителни набори (Ho et al., 1994; Kittler et al., 1998). Съществена характеристика на многостепенния класификатор е фактът, че той може да стабилизира обучението на класификатори въз основа на малък размер на извадката.

Псевдокодът на хибридния алгоритъм H1, базиран на агрегирането на NB и KNN, е следният:

```
Step 1: Aggregate the probabilities from NB and KNN
  For each test instance Ti
    For each class label Ci
      Probability of Ci= (probability of Ci obtained from NB + probability of Ci
obtained from KNN)/2
    End for
  End for
Step 2: Assign the class label with highest probability to the test instance.
```

Псевдокодът на хибридния алгоритъм H2, базиран на агрегирането на NB и RF, е следният:

```
Step 1: Aggregate the probabilities from NB and RF
  For each test instance Ti
    For each class label Ci
    Probability of Ci= (probability of Ci obtained from NB + probability of Ci
obtained from RF)/2
    End for
  End for
Step 2: Assign the class label with highest probability to the test instance.
```

Псевдокодът на хибридния алгоритъм H3, базиран на агрегирането на NB, RF и KNN, е следният:

```
Step 1: Aggregate the probabilities from NB, RF and KNN
  For each test instance Ti
    For each class label Ci
    Probability of Ci= (probability of Ci obtained from NB + probability of Ci
obtained from RF + probability of Ci obtained from KNN)/3
    End for
  End for
Step 2: Assign the class label with highest probability value to the test
instance.
```

Предлага се среднопретеглената точност при комбиниране на резултати от различни алгоритми за машинно обучение да се изрази чрез следното съотношение:

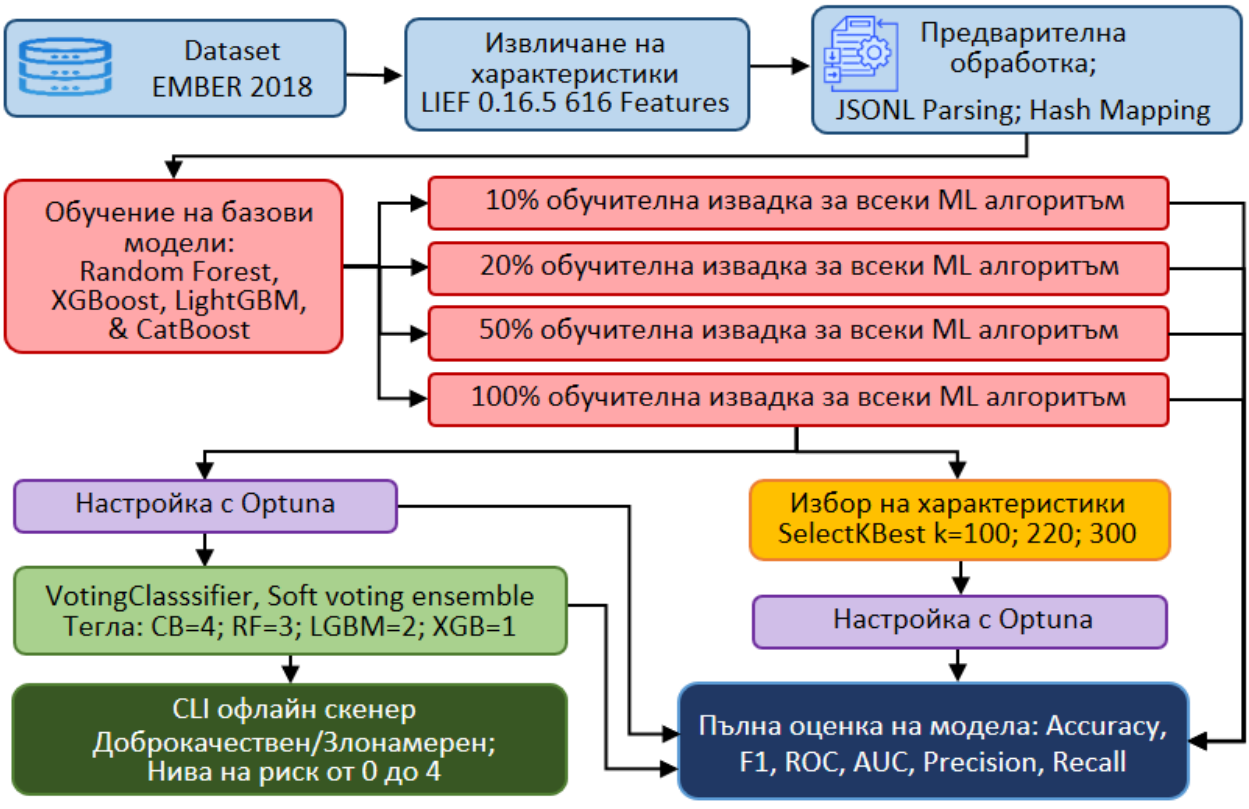
$$Conbined\_Accuracy = (\sum_{i=1}^{n} Accuracy_i * Count_i)(\sum_{i}^{n} Count_i)^{-1} \quad (2.10)$$

където $Accuracy_i$ точността на i-тия алгоритъм докато $Count_i$ представлява броя пъти, в които е постигната i-тата точност (т.е. броят на алгоритмите, които са довели до i-тата точност), а n е общият брой уникални точности.

2.3. Рамка за статична класификация на зловреден софтуер, използваща оптимизация на функции и ансамблово обучение

2.3.1. Методология на Python-базирана софтуерна рамка за класификация на статичен зловреден софтуер, използваща оптимизация на характеристики и ансамбълно обучение

Предложена е методология за реализиране на рамка за откриване и класифициране на зловреден софтуер чрез използването на Python библиотеки. Тази рамка комбинира няколко ML алгоритъма, като Random Forest, XGBoost, LightGBM и CatBoost (Thai, 2022; Hancock & Khoshgoftaar, 2020). Предложената рамка разчита на модулна архитектура и може лесно да добавя допълнителни модули с различни ML алгоритми. При разработването на Python-базирана рамка за автоматична класификация на зловреден софтуер се предлага следната методология, илюстрирана на Фиг. 2.6.



Фиг. 2.6. Методология на разработената Python-базирана рамка за класификация на статичен зловреден софтуер

2.3.2. Обучение на моделите и резултати

За всеки експеримент за всеки алгоритъм (по един за всеки тренировъчен набор) се обучават четири модела, което води до 16 базови модела. Показателите за оценка на тези 45 обучени модела са показани в Таблица 2.2.

Таблица 2.2. Резултати от моделите с различни тренировъчни набори от данни.

#	Model Name	K	Accuracy	F1-Score	ROC AUC	Precision	Recall	Optuna
	100% Train Size							
1	RandomForest	616	0.9391	0.9392	0.9796	0.9353	0.9433	Yes
2	RandomForest	616	0.9245	0.9241	0.9542	0.9285	0.9198	No
3	RandomForest	300	0.9117	0.9116	0.9426	0.9129	0.9093	No
4	RandomForest	300	0.9362	0.9369	0.9777	0.9332	0.9387	Yes
5	RandomForest	200	0.9371	0.9371	0.962	0.9344	0.9394	Yes
6	RandomForest	200	0.9188	0.9184	0.9435	0.9193	0.9175	No
7	RandomForest	100	0.9357	0.9356	0.9506	0.9328	0.9382	Yes
8	RandomForest	100	0.9154	0.9154	0.9454	0.9161	0.9145	No
9	XGBoost	616	0.9173	0.9184	0.9728	0.9063	0.9308	Yes
10	XGBoost	616	0.9082	0.9171	0.9482	0.8926	0.9281	No
11	XGBoost	300	0.9283	0.9287	0.9586	0.9207	0.9368	Yes
12	XGBoost	300	0.7762	0.8027	0.9124	0.7176	0.9106	No
13	XGBoost	200	0.9268	0.9269	0.9735	0.9201	0.9336	Yes
14	XGBoost	200	0.7604	0.7781	0.9007	0.7246	0.8412	No
14	XGBoost	100	0.9255	0.9258	0.9407	0.9191	0.9324	Yes
15	XGBoost	100	0.8294	0.8215	0.8394	0.8616	0.7852	No
16	LightGBM	616	0.9389	0.9391	0.9691	0.9351	0.9432	Yes
17	LightGBM	300	0.9441	0.9438	0.9853	0.9445	0.9437	Yes
18	LightGBM	200	0.9424	0.9425	0.9841	0.9399	0.9451	Yes
19	LightGBM	100	0.9427	0.9425	0.9576	0.9432	0.9418	Yes
20	LightGBM	200	0.9011	0.9036	0.9283	0.8815	0.9268	No
21	LightGBM	616	0.8970	0.8993	0.9645	0.8795	0.9201	No
22	LightGBM	300	0.8257	0.8408	0.9232	0.7739	0.9203	No
23	CatBoost	616	0.9405	0.9406	0.9706	0.9388	0.9424	Yes
24	CatBoost	616	0.9292	0.9298	0.9597	0.9215	0.9382	No
25	CatBoost	300	0.9341	0.9342	0.9757	0.9299	0.9385	Yes
26	CatBoost	300	0.8447	0.8503	0.8795	0.8206	0.8823	No
27	CatBoost	200	0.9355	0.9359	0.9608	0.9307	0.9411	Yes
28	CatBoost	200	0.9237	0.9236	0.9689	0.9246	0.9226	No
29	CatBoost	100	0.9315	0.9313	0.9464	0.9275	0.9351	Yes
30	CatBoost	100	0.8507	0.8517	0.8665	0.8457	0.8578	No
31	VotingClassifier (XGB, LGBM, CB, All Features)	616	0.9461	0.9461	0.9864	0.9468	0.9454	No

	50% Train Size							
32	RandomForest	616	0.8652	0.8533	0.9253	0.9338	0.7856	No
33	XGBoost	616	0.8668	0.8621	0.9216	0.8934	0.8329	No
34	LightGBM	616	0.8659	0.8605	0.9352	0.8967	0.8272	No
35	LightGBM	100	0.8827	0.8841	0.8989	0.8735	0.8951	No
36	CatBoost	616	0.8891	0.8846	0.9543	0.9223	0.8498	No
	20% Train Size							
37	RandomForest	616	0.7855	0.7375	0.9247	0.9497	0.6028	No
38	XGBoost	616	0.8381	0.8244	0.9256	0.9003	0.7603	No
39	LightGBM	616	0.8349	0.8141	0.9244	0.9314	0.7234	No
40	CatBoost	616	0.8625	0.8487	0.9408	0.9433	0.7713	No
	10% Train Size							
41	RandomForest	616	0.5972	0.3336	0.7517	0.9663	0.2016	No
42	XGBoost	616	0.6626	0.5165	0.8121	0.9112	0.3604	No
43	LightGBM	616	0.7463	0.6683	0.9034	0.9658	0.5112	No
44	CatBoost	616	0.6458	0.4664	0.7613	0.9458	0.3096	No

2.3.3. Избор на характеристики и оптимизация на хиперпараметри.

Прилагат се два важни метода за оптимизиране и систематизиране на производителността: намаляване на размерността чрез SelectKBest и оптимизация на хиперпараметри с помощта на Optuna. Общо 45 модела са обучени и оценени със следните комбинации: 4 алгоритъма за машинно обучение (Random Forest, XGBoost, LightGBM, CatBoost), 4 различни размера на тренировъчния набор (10%, 20%, 50%, 100%), Optuna включена и изключена и различни стойности на SelectKBest, k = 100; 200; 300; всеки модел се оценява от независим тестов набор по 5 основни показателя (вж Таблица 2.2).

Алгоритъмът за избор на характеристики SelectKBest, използващ mutual_info_classif, се използва за идентифициране на най-информативните характеристики. При сравняване на броя характеристики и производителността, параметърът k се тества съответно със 100, 200 и 300 характеристики. Има малка загуба на точност, дължаща се на по-малко избрани характеристики, но както k=200, така и k=300 не се различават много от целия набор от характеристики, което означава, че по-малко функции могат да бъдат полезни в ситуации, където изчислителните ресурси са фактор. Пълното оптимизиране на различни модели с Optuna изисква значително количество изчислителни и времеви ресурси: всеки от 4-те алгоритъма (CatBoost, XGBoost, LightGBM, Random Forest) е изпълнил 50 опита плюс различни измерения на характеристиките и дълбочини на модела, което е отнело приблизително от 100 до 140 часа чисто изчислително време.

2.3.4. Оптимизирани хиперпараметри от най-добре представящите се конфигурации

Хиперпараметрите са оптимизирани въз основа на 50-те пробни Optuna процеса, извършени върху всеки от моделите. Те са най-добрите конфигурации, базирани на F1-

оценка на валидационния набор, а крайните сравнения на моделите са използвали тези стойности. Описаните хиперпараметри са добре балансирани мерки за производителност/скорост и са валидирани с помощта на пълния набор от функции (616 функции) с трансформации SelectKBest (k=100;200;300).

2.3.5. Ансамблов модел с класификатор за гласуване (Voting Classiffier)

Създаден е ансамблов модел, който комбинира оптимизирани модели с уникални архитектури, даде най-благоприятния резултат в анализа. Моделът VotingClassifier, който прилага меко гласуване (Soft Voting) с определени тегла, направи прогнози по-силни от отделните модели, включително CatBoost, който имаше най-добра производителност. Това е доказателство за оползотворяване на силата на уникални модели с уникални полезни алгоритми.

Съставът на VotingClassifier е следният: CatBoost (Optuna, 616 характеристики); LightGBM (Optuna, 616 характеристики); Random Forest (Optuna, 616 характеристики); и XGBoost (Optuna, 616 характеристики). Мекото гласуване (Soft Voting) се използва за обединяване на вероятностни резултати от модела. Мекото гласуване е предпочитано пред твърдото гласуване, защото то позволява обединяването на прогнозите по много по-информативен начин, тъй като прогнозираните вероятности могат да бъдат обединени и разгледани, а не само обединени и твърдо определени класификации.

2.4. Самоосъзната класификация на зловреден софтуер чрез рутиране на модели на базата на система за доверие за избора и обяснимост на характеристиките

2.4.1. Набор от данни и извличане на характеристики

EMBER 2024 е широкомащабен отворен набор от данни за откриване на статичен зловреден софтуер, разработен от Future Computing Lab и публикуван в началото на 2024 г. EMBER 2024 е официалният наследник на бенчмарк набора от данни EMBER 2018. EMBER 2024 включва общо над 5 милиона примера, в много етикети, включително:

- 4 000 000 примера за обучение (Win32, .NET и неизвестни PE)
- 1 000 000 тестови примера (Win32 и други PE)
- Множество типове класове/архитектура, включително специфични за .NET и подписани с Authenticode файлове

Извличането на характеристики е извършено с помощта на модифициран PEFeatureExtractor конвейер с lief==0.16.5 и Python 3.8, за да се осигури ретросъвместимост със стари PE формати. Всяка извадка се извлича в 616-измерен вектор, който включва байтови хистограми, ентропийни части и низови характеристики. Таблица 2.3 илюстрира основните разлики между наборите от данни EMBER 2018 и EMBER 2024.

Таблица 2.3. EMBER 2018 и EMBER 2024 набори от данни

Attribute	EMBER 2018	EMBER 2024 (our sample)
Release year	2018	2024
Training samples	900 000	1 560 000
Test samples	200 000	360 000
Feature count	616	616 (harmonized)

Регистрираните най-добрите параметри на всеки модел са оценени, използвайки различни показатели (точност, F1-оценка, ROC-AUC и прецизност, попълнота), представени в Таблица 2.4:

Таблица 2.4. Сравнение на производителността на отделни класификатори, оптимизирани за Optuna варианти и ансамбъла VotingClassifier върху EMBER 2024 (Win32)

Model	Accuracy	F-1 score	ROC-AUC	Precision	Recall
CatBoost	0.9397	0.9382	0.9881	0.9622	0.9153
LightGBM	0.9438	0.9425	0.9898	0.9650	0.9210
LightGBM (Optuna)	0.9540	0.9531	0.9925	0.9710	0.9359
RandomForest	0.9545	0.9535	0.9927	0.9746	0.9333
CatBoost (Optuna)	0.9546	0.9538	0.9924	0.9710	0.9372
RandomForest (Optuna)	0.9552	0.9542	0.9928	0.9747	0.9346
XGBoost	0.9575	0.9568	0.9936	0.9730	0.9411
XGBoost (Optuna)	0.9664	0.9659	0.9955	0.9795	0.9527
VotingClassifier (All models)	0.9654	0.9649	0.9953	0.9793	0.9509

Ансамблов модел: VotingClassifier

Създаден е ансамблов модел с помощта на две комбинации от оптимизираните класификатори чрез VotingClassifier, с меко гласуване и тегла за всеки класификатор в зависимост от неговата надеждност.

(1) Първа комбинация: XGBoost: 10; RandomForest: 2; CatBoost: 1; LightGBM: 4

(2) Втора комбинация: XGBoost: 15; RandomForest: 1; LightGBM: 3

Въпреки че се постарахме да изградим силен алгоритъм на VotingClassifier, оптимизираният алгоритъм XGBoost превъзхожда другите алгоритми по отношение на обобщаване и е основният класификатор в SAMC рамката.

Самоосъзнаващ се класификатор на модели)

Разработен е самоосъзнаващ се класификатор на модели (SAMC), който динамично да насочва към подходящия модел въз основа на нивото на доверие на прогнозата. Целта на логиката на SAMC е да намери баланс между доверие, обобщаемост и устойчивост при използване на стари и съвременни модели на машинно обучение, като и двата модела са обучени върху два различни набора от данни: EMBER 2018 (стар) и EMBER 2024 (нов).

Всеки .exe файл първо се анализира статично чрез PEFeatureExtractor пайплайн, което води до 616-измерен вектор от характеристики. След това векторът се въвежда в следните два класификатора, които са обучени независимо:

- Наследен (Legacy) модел: VotingClassifier трениран върху EMBER 2018, който е съставен от моделите RandomForest, CatBoost, LightGBM, и XGBoost с меко гласуване (тежести: RF=3, CB=2, LGBM=1, XGB=4), постигайки следните най-добри показатели върху legacy данните: Accuracy 0.9646 0.9461, F1-Score 0.9646 0.9461, ROC-AUC 0.9951 0.9864, Precision 9468 , Recall 0.9454
- Нов (New) модел: един XGBoost (Optuna) класификатор трениран върху 2024 Win32 проби от EMBER. Той също така постига всички най-добри показатели: Accuracy 0.9583 0.9664, F1-Score 0.9576 0.9659, and ROC-AUC 0.9938 0.9527, Precision 0.9795, Recall 9527.

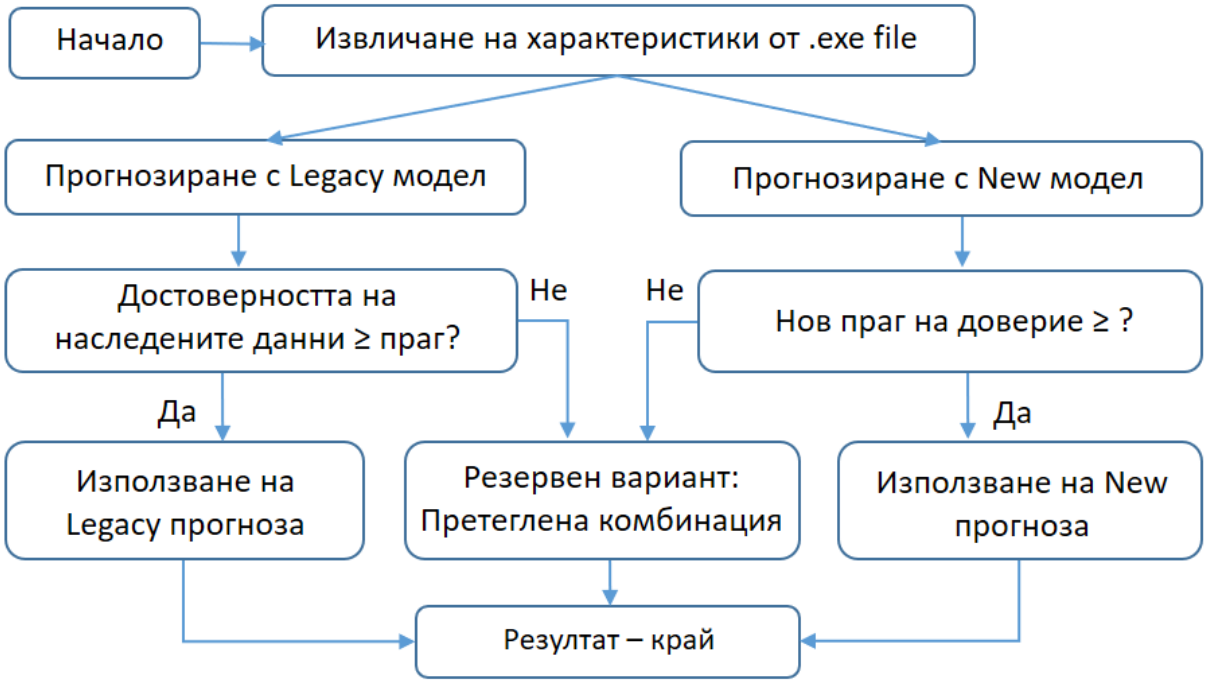
Нека maxproba означава максималната прогнозирана вероятност за клас на даден модел. SAMC може да представи уравнение, което да определи рутиране въз основа на следната логика:

- Ако max_proba(new_model) ≥ 0.85: довери се на прогнозата на новия модел,
- Ако max_proba(legacy_model) ≥ 0.85: довери се на прогнозата на стария модел,
- В противен случай: комбинирай прогнозираните вероятности за класове и на двата модела като резервен вариант, като предложите среднопретеглена стойност, както следва:

$$P_{final} = 0.6 \cdot P_{new} + 0.4 \cdot P_{legacy}$$

Най-високата вероятност от тази крайна вероятност ще бъде избрана като окончателен резултат, взет от SAMC. Механизмът за рутиране, съобразено с доверието, в рамките на SAMC гарантира, че прогнозите ще бъдат приемани само когато и двата отделни модела са достатъчно уверени, ограничавайки вероятността от приемане на прекалено уверени погрешни класификации на нови или двусмислени нови данни. Логиката на рутиране на SAMC е позакана на Фиг. 2.7:

В резултат на проведено изследване се установи статистически значимо отклонение по много характеристики. В Таблица 2.6 са обобщени резултатите за 10-те характеристики с най-висока статистика на KS теста:



Фиг. 2.7. Логика на рутиране на SAMC с избор, базиран на доверие, и претеглена резервна стратегия

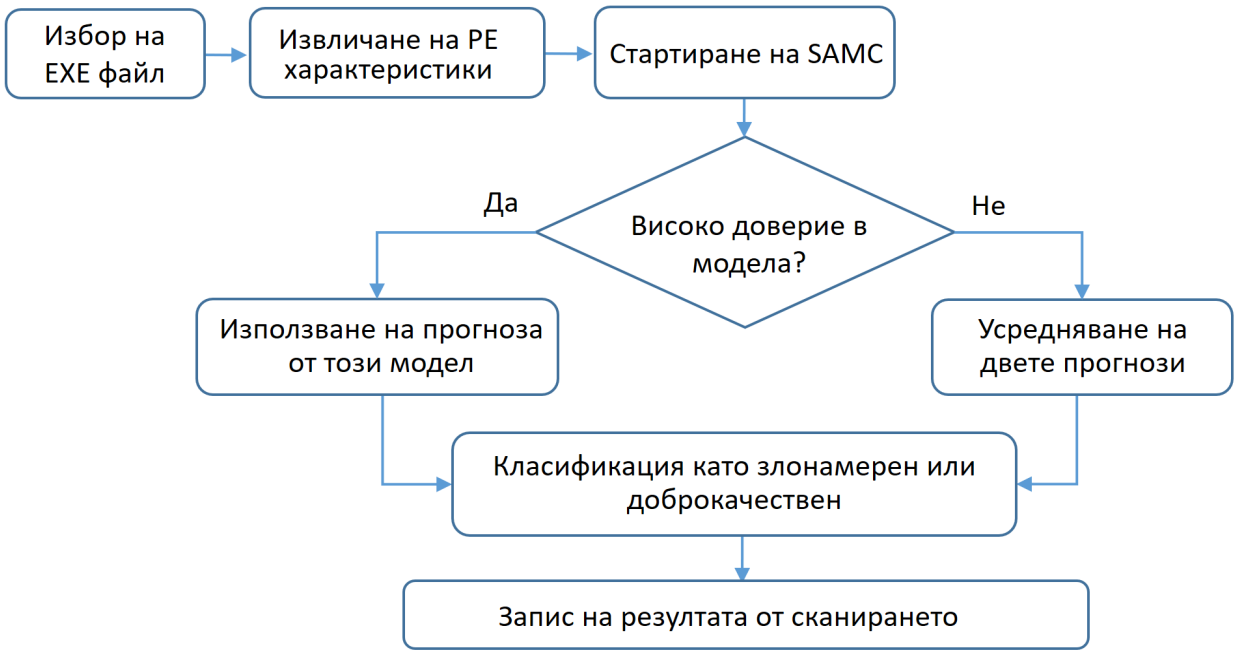
Таблица 2.6. Топ 10 характеристики със статистика за тестове на KS

Feature Index	KS Statistic	p-value
Feature 501	0.3983	< 1e-10
Feature 507	0.3976	< 1e-10
Feature 506	0.3971	< 1e-10
Feature 499	0.3959	< 1e-10
Feature 508	0.3959	< 1e-10
Feature 511	0.3959	< 1e-10
Feature 510	0.3958	< 1e-10
Feature 498	0.3950	< 1e-10
Feature 505	0.3945	< 1e-10
Feature 509	0.3943	< 1e-10

2.4.3. Скенер за самоосъзната класификация на зловреден софтуер с графичен потребителски интерфейс

За да се подобри прозрачността, достъпността и обяснимостта на предложените модели за класификация на зловреден софтуер е необходимо да създадем графичен потребителски интепфейс. Той трябва да позволява на потребителя да сканира изпълним файл, използвайки три режима за избор: (1) Manual Legacy Mode: използва оригиналния модел, обучен на EMBER 2018; (2) Manual New Model Mode: използва оптимизирания модел, обучен на EMBER 2024; (3) Auto Mode (SAMC): извършва интелигентен избор на модел въз основа на прагове на доверие.

На Фиг. 2.16 е представена илюстрация на интегрирания поток на рутиране, така както е планирано да се реализира чрез графичния потребителски интерфейс.



Фиг. 2.16. Сканиране и рутиране с помощта на графичен потребителски интерфейс, включително логика за резервен подход, базирана на доверие

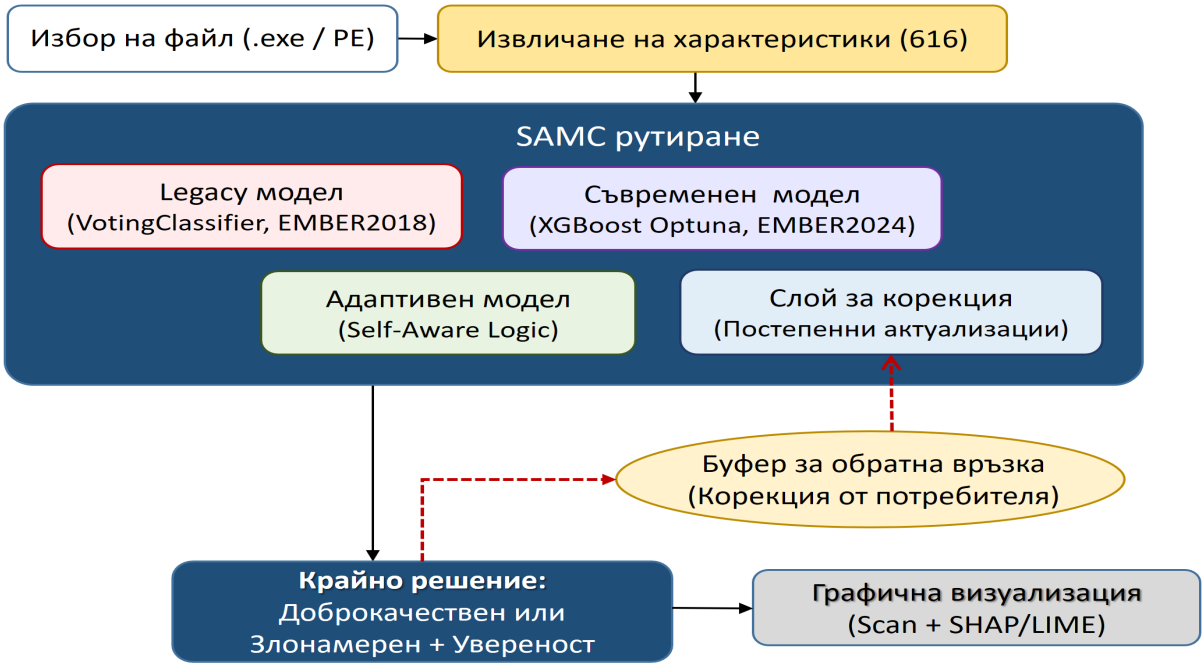
След като потребителят избере файл, той влиза в SAMC, след като статичните характеристики бъдат извлечени. SAMC определя кой модел (стар или нов) трябва да се използва въз основа на това колко уверена е била някоя от прогнозите или, ако някоя от прогнозите не е представила значимо ниво на увереност, каква тежест е била предоставена като резервен вариант. Системата ще информира за решението и ще предостави SHAP-базирано обяснение, ако е бил използван новият модел.

2.5. Адаптивна рамка, интегрираща доверие за класификация на зловреден софтуер чрез корекции за обратна връзка

2.5.1. Проектиране и внедряване на системата

Адаптивната рамка за класификация на зловреден софтуер с обратна връзка, Shipka Guard, е реализирана чрез интегриране на наследени и съвременни модели с адаптивен слой, базиран на обратна връзка. Системата е организирана в няколко взаимосвързани компонента, които са илюстрирани на Фиг. 2.17.

Рамката следва пайплайн дизайн, при който качените PE файлове се обработват чрез извличане на характеристики, избор на модел (SAMC рутиране), опционално пачване на корекции и вземане на решения. Потребителската обратна връзка се събира непрекъснато и интегрира, за да се подобри устойчивостта на модела с течение на времето.



Фиг. 2.17. Компоненти на Shipka Guard, показващи извличане на функции, SAMC рутиране, буфер за обратна връзка, слой за корекции и интеграция с графичен потребителски интерфейс

Системата комбинира три вида модели:

(1) Наследен модел - Legacy model (обучен върху EMBER 2018);

(2) Модерен модел - Modern model (обучен върху EMBER 2024); and

(3) Адаптивен модел - Adaptive model (периодично преобучаван чрез обратна връзка).

Прогнозите се рутират с помощта на SAMC като:

- Ако даден модел прогнозира 0,85 или по-висока стойност, тогава той се избира за прогнозиране.
- В противен случай се използва резервния ансамбъл с тегла: Adaptive model 0,5, Modern model 0,3, Legacy model 0,2.

За да се осигури гъвкавост и сравнимост в различните среди, предложената системата трябва поддържа четири режима на сканиране:

(1) Автоматичен – Auto (SAMC): рутиране, съобразено с доверието, с праг на доверие и резервен ансамбъл.

(2) Форсиран Адаптивен – Force Adaptive: налага използването на най-новия адаптивен модел.

(3) Форсиран Модерен – Force Modern: налага използването на модерния модел.

(4) Форсиран Наследен – Force Legacy: Принудително наследяване: налага използването на наследения модел.

Режимите поддържат както автоматизирано поведение, съобразено с доверието, така и ръчно сравнение между поколения модели.

Слоят за корекция е лек SGDClassifier, който се обучава постепенно върху потребителската обратна връзка. Неговата важност се мащабира динамично като функция

на уникалните записи за обратна връзка, което гарантира, че поведението му при корекция остава стабилно при малки размери на извадката и с много валидиращи корекции теглото му се увеличава с течение на времето. По-конкретно, корекцията расте по формулата:

$$w_{patch} = min\left(\alpha, \frac{n}{n+k}\alpha\right) \tag{2.11}$$

където n е броят на уникалните проби за обратна връзка, α е максималният принос за избраната предварително зададена настройка, а k е константа на изглаждане, контролираща скоростта на растеж.

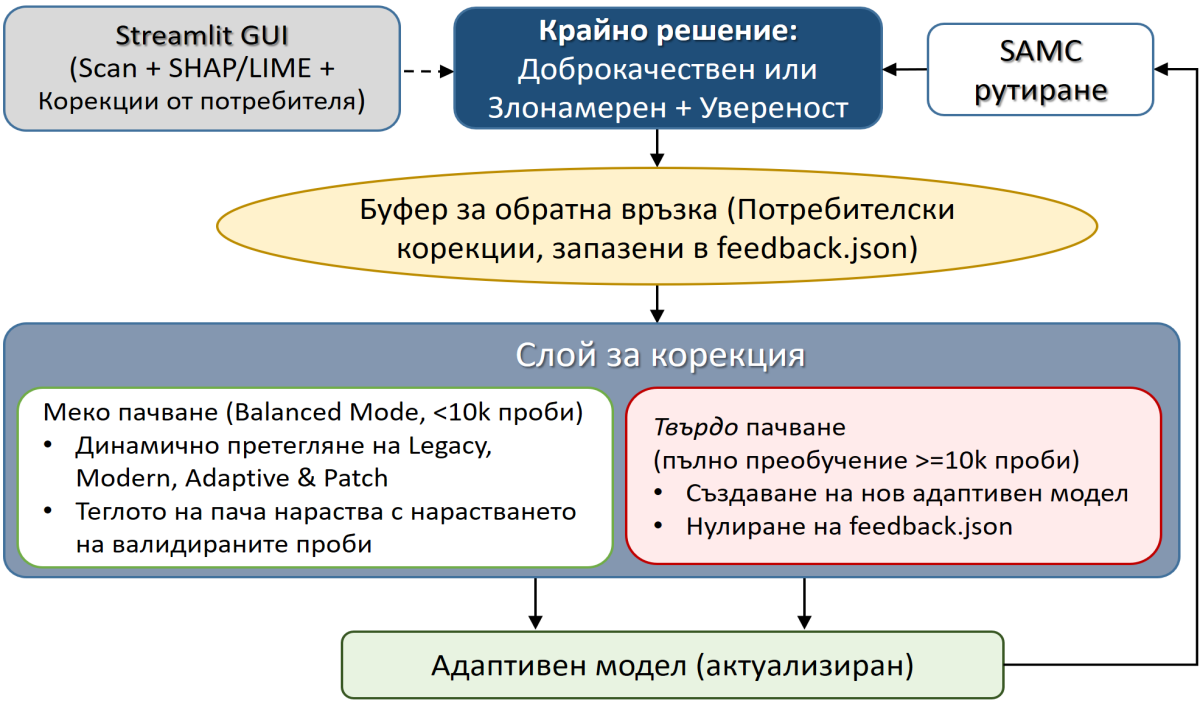
Поддържат се три оперативни предварително зададени настройки:

(1) Консервативна: $\alpha = 0.12, k = 300$;

(2) Балансирана: $\alpha = 0.20, k = 100$; and

(3) Агресивна: $\alpha = 0.30, k = 50$.

Предвидена е и опция за синтетично FN/FP инжектиране, което помага при оценката на устойчивостта и стрес тестването на механизма на пача преди мащабно внедряване. Този дизайн позволява на пач слоя да има незначително влияние, когато са налични само няколко проби за обратна връзка, като същевременно бавно увеличава влиянието си с набирането на по-стабилни корекции (Фиг. 2.18).



Фиг. 2.18. Адаптация на пач слоя: меко пачване (претегляне, управлявано от обратна връзка) спрямо твърдо пачване (пълно преобучение с ≥10k проби)

След като буферът за обратна връзка достигне до конфигурируемия праг (≥ 10 000 проби), системата инициира пълно преобучение на адаптивния модел в XGBoost. Наборът от данни за обратна връзка след това се експортира във формат JSONL за целите на възпроизводимостта и на моделът му се дава версия в директорията с модели.

ГЛАВА 3. ЧИСЛЕНО ТЕСТВАНЕ НА ПРЕДЛОЖЕНИТЕ МОДЕЛИ ЗА ИЗБОР НА ВИРТУАЛНА МАШИНА, ХИБРИДНИ АЛГОРИТМИ И РАМКИ ЗА ОТКРИВАНЕ НА ЗЛОНАМЕРЕН СОФТУЕР

3.1. Числено тестване на моделите за групово вземане на решение при избор на софтуер за виртуални машини

За целите на анализа и откриването на зловреден софтуер е необходимо да се инсталира десктоп софтуерът за виртуални машини на Windows. За целите на числените тестване е използвана публикувана информация за вече оценени софтуерни продукти, които са показани в Таблица 3.1.

Таблица 3.1. VM Software for Desktop Windows Deployment

VM Software	Review Rating			
	Ease of Use	***Customer Service***	***Features***	***Value for Money***
Microsoft Azure	4.1	4.2	4.6	4.2
VirtualBox	4.3	4.1	4.5	4.7
NAKIVO Backup & Replication	4.8	4.8	4.7	4.7
Altaro VM Backup	4.8	4.8	4.6	4.7
DiskStation	4.6	4.2	4.7	4.6
Ahsay Offsite Backup Server	3.8	3.4	4.0	3.7
Uranium Backup	4.7	4.7	4.5	4.7
Comet Backup	4.4	4.6	4.5	4.6
VMware Workstation Pro	4.6	4.6	4.7	4.2
Iperius Backup	4.1	4.1	4.1	4.3

Note: Data are taken from:
https://www.capterra.com/virtual-machine-software/?sortOrder=most_reviews&platform=%5B4%5D

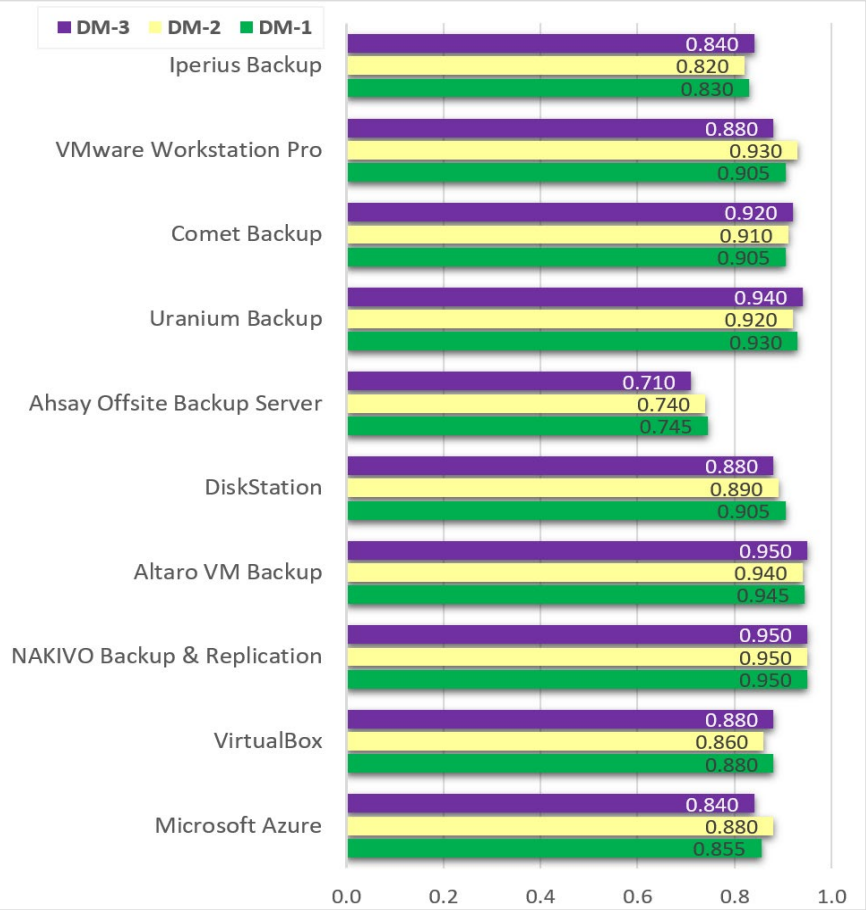
3.1.2. Резултати от числените тестване

За да се тества ефективността на предложените модели за избор на софтуер за виртуални машини, група от трима експерти, формиращи групата са представили своите мнения относно важността на четирите критерия: 1) лекота на използване; 2) обслужване на клиентите; 3) характеристики; и 4) съотношение цена-качество. Оценките на тези критерии, показани в Таблица 3.1 са нормализирани в интервала от 0 до 1, за да се получи сравним диапазон с други коефициенти, използвани във формулираните модели. Коефициентите, които изразяват важността на критериите според мненията на водещите мениджъри, са показани в Таблица 3.2.

Таблица 3.2. Предпочитания на експертите по отношение на критериите за оценка

Мениджъри	Коефициенти, изразяващи важността за критериите			
	Ease of Use	***Customer Service***	***Features***	***Value for Money***
DM-1	0.25	0.25	0.25	0.25
DM-2	0	0.50	0.50	0
DM-3	0	0.50	0	0.50

Използвайки данните от Таблица 3.2, формулирания оптимизационен модел (2.1) – (2.3) и нормализираните оценки в интервала от 0 до 1, може да се определи общото представяне на всички алтернативи в съответствие с различните гледни точки на членовете на групата, както е показано на Фиг. 3.1.



Фиг. 3.1. Общо представяне на всички алтернативи.

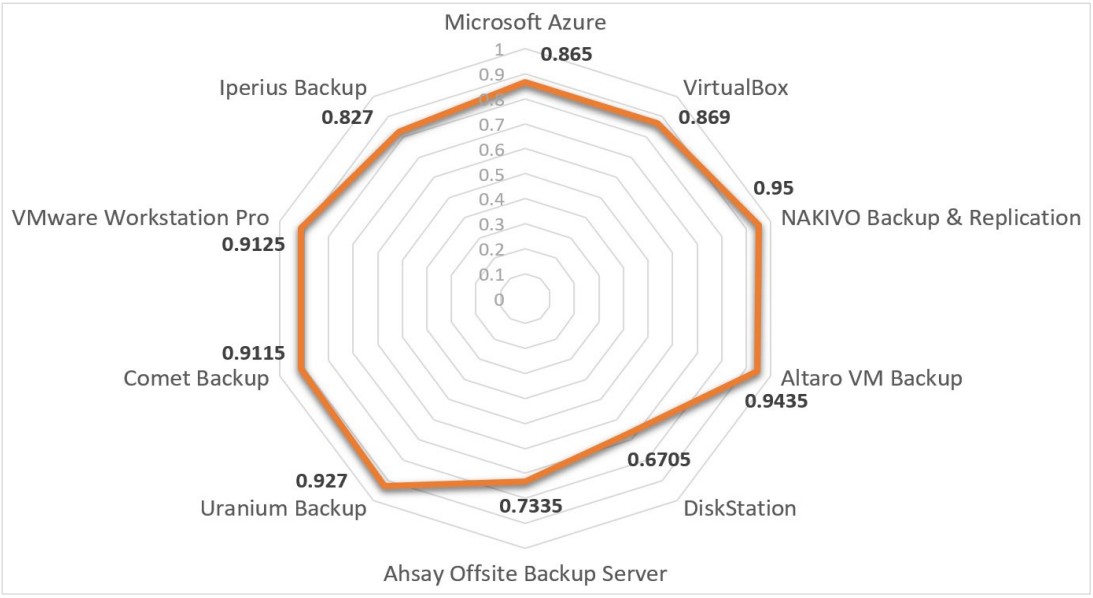
Интересно е да се покаже как крайното групово решение се влияе, когато се използват допълнителни коефициенти за мненията на членовете на групата, показани в Таблица 3.3.

Таблица 3.3. Тегла за важност на мнението на експертите

Тегла за експертите при формиране на окончателното групово решение		
DM-1	***DM-2***	***DM-3***
0.20	0.55	0.25

Използвайки данните от Таблица 3.1, Таблица 3.2 и Таблица 3.3, заедно с предложения модел за групово вземане на решения (2.1) – (2.3), се получава следното крайно класиране на алтернативите, както е показано на Фиг. 3.2.

Формираното крайно групово решение за избора е съставено от предпочитанията на всичките членове на групата, заедно с допълнителните коефициенти от Таблица 3.3, определя „NAKIVO Backup & Replication“ като най-добър избор, тъй като неговото представяне има най-висока стойност, равна на 0.95.



Фиг. 3.2. Класиране на алтернативите, отчитайки груповото решение.

Възможно е изборът да се прецизира чрез намаляване на множеството от алтернативи, от които да се избира чрез използване на предложения втори оптимизационен модел за групово вземане на решение (2.4) – (2.9). При този модел се формулиране оптимизационна задача, чието решение определя едновременно повече от една алтернатива, редуцирайки първоначалното множество от алтернативи. Едновременното определяне на 3, респективно на 5 алтернативи е показано в Таблица 3.4.

Таблица. 3.4. Стойности на променливите при избор на множество алтернативи

VM software	Избор на 3 алтернативи	Избор на 5 алтернативи
Microsoft Azure	0	0
VirtualBox	0	0
NAKIVO Backup & Replication	1	1
Altaro VM Backup	1	1
DiskStation	0	0
Ahsay Offsite Backup Server	0	0
Uranium Backup	1	1
Comet Backup	0	1
VMware Workstation Pro	0	1
Iperius Backup	0	0

Предложените два математически модела реално показват, че могат да се използват успешно за избор на софтуер за виртуална машина. Чрез първия модел може да се определи най-добрата алтернатива, като вземе предвид оценките по критериите, важността на критериите и коефициенти за важност на мненията на членовете на групата. При втория модел могат да се определят повече от една алтернатива решавайки съответната оптимизационна задачата, благодарение на използването на двоични целочислени променливи.

3.2. Тестване на предложения подобрен подход за статичен анализ за откриване на зловреден софтуер чрез оптимизиране на извличането на характеристики, комбинирайки различни алгоритми за машинно обучение

3.2.1. Изходни данни

Наборът от данни IoT-23, генериран от лабораторията Avast AIC с помощта на Zeek Network Security Monitor, съдържа прихванати записи с доброкачествен и злонамерен трафик от 2018 до 2019 г. Наборът от данни включва пълни и по-леки версии. Той обозначава различни типове атаки като „Доброкачествена“, „DDoS“, „Okiru“ и „Част от хоризонтално сканиране на портове“. Интересно е да се проучи дали предложените хибридни алгоритми биха могли да подобрят точността при откриване на злонамерен софтуер. За целта се използва подмножество от набора от данни IoT-23, състоящо се от 500 000 екземпляра, което включва както доброкачествени, така и злонамерени класове. Експериментът се провежда, следвайки 5 основни етапа, както е показано на Фиг. 3.4:

Фиг. 3.4. Работен процес на експеримента

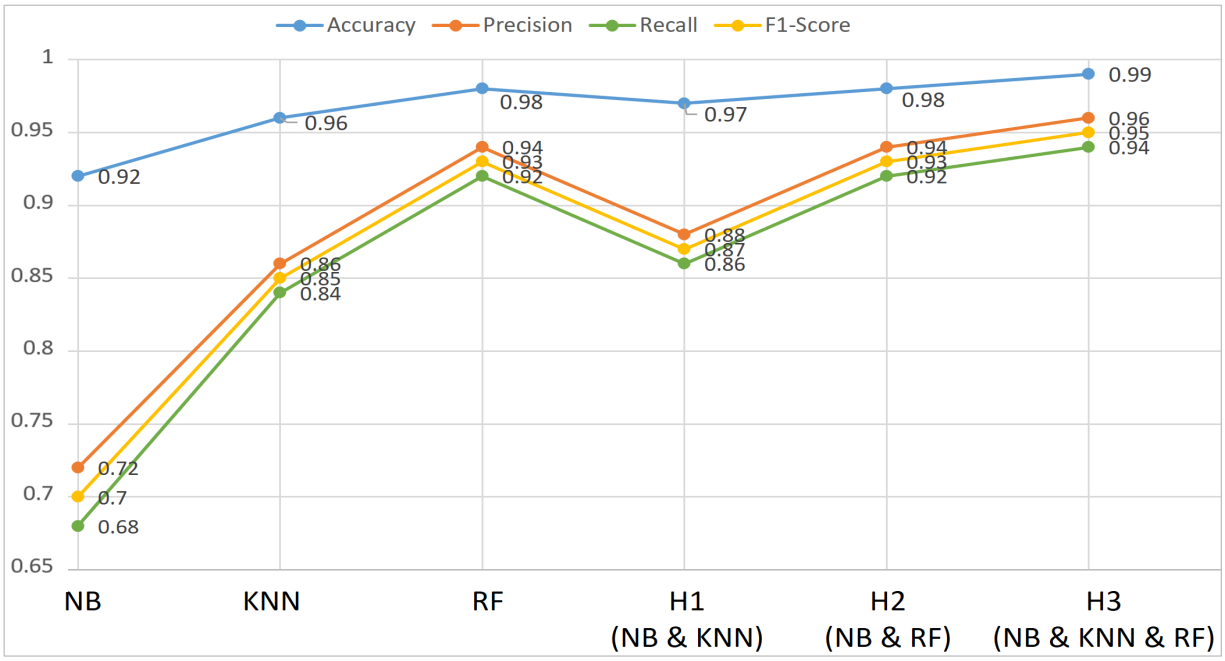
3.2.2. Анализ и обсъждане на резултати

Резултатите от производителността, използващи единична имплементация на алгоритмите, и предложените хибридни алгоритми са показани в Таблица 3.5.

Таблица 3.5. Резултати от производителността на алгоритмите

Model	Accuracy	Precision	Recall	F1-Score
NB	0.92	0.72	0.68	0.70
KNN	0.96	0.86	0.84	0.85
RF	0.98	0.94	0.92	0.93
H1 (NB & KNN)	0.97	0.88	0.86	0.87
H2 (NB & RF)	0.98	0.94	0.92	0.93
H3 (NB & KNN & RF)	0.99	0.96	0.94	0.95

Както се вижда от Таблица 3.5, някои от комбинираните алгоритми показват по-добра производителност. Сравнението между всички тези показатели и различните модели и комбинации от моделите е показано на Фиг. 3.5.



Фиг. 3.5. Сравнение на показателите на различни алгоритми и техните комбинации

Предложените хибридни алгоритми допринасят значително за подобряване на производителността. Получените резултати в следствие на проведените експерименти подчертават значението на оптимизирането на извличането на характеристики, което е от решаващо значение за повишаване на точността на моделите за машинно обучение. Ефективното извличане на характеристики трансформира суровите данни във формат, който е по-информативен и дискриминативен за задачата, подобрявайки процеса на обучение.

3.3. Числено тестване на предложената рамка за статична класификация на зловреден софтуер, използваща оптимизация на функции и ансамблово обучение

Извличането на характеристики се извършва чрез модифициране на съществуващия Python код за характеристики от проекта EMBER, за да работи с lief 0.16.5 и Python 3.8. Стойността на ентропията, имената на секциите, елементите за импортиране/експортиране и отчетените статистически данни за метаданни, низове и др. Всеки PE файл е представен като вектор от 616 променливи.

3.3.1. CLI скенер: Практически случай на употреба

CLI скенерът е създаден специално за използване в среда (Python 3.8 и lief 0.16.5), за да се гарантира пълна съвместимост с извличането на характеристики на EMBER 2018 и за да се предотвратят проблеми с по-нови PE файлови структури. CLI скенерът предоставя среда от команден ред, която използва входен аргумент, предоставен от потребителя (пътят до PE файла), за да извлече характеристики (с помощта на вграден механизъм), да зареди крайния модел VotingClassifier и да получи ниво на риск. Разработеният CLI скенер е напълно офлайн, така че може да работи в изолирани мрежи и среди (т.е. виртуални). Той

е създаден в Python среда и е пакетирано .exe приложение, използващо PyInstaller. Характеристиките се извличат с помощта на lief и са точно 616. CLI скенерът е способен да генерира и поддържа лог файл, показващ подробно състояние на сканирането и вероятностни оценки.

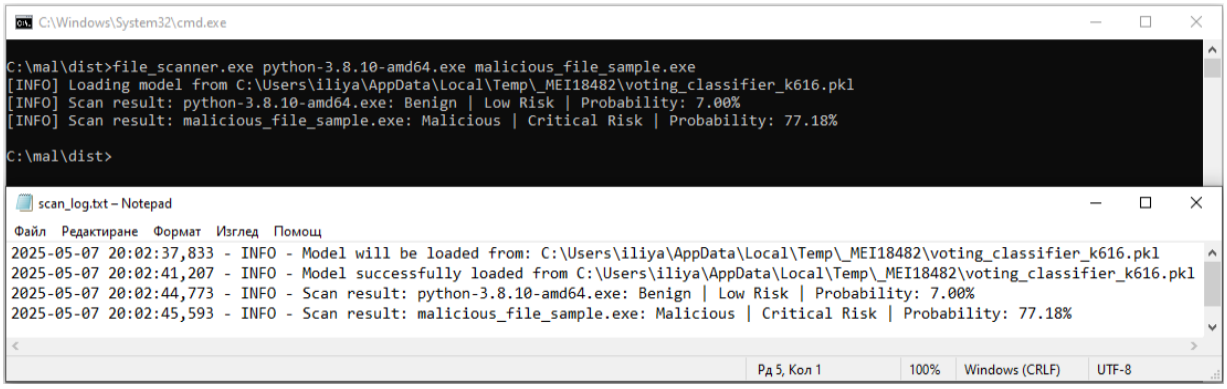
Производителността на CLI скенера е следната: Средното време за сканиране на един PE файл е ~1,0 секунди. Всеки конзолен изход се записва и регистрира, което включва всички подробности, както бихте видели в конзолния изход: дата и час, път на файла, прогнозирана вероятност, рискова оценка и краен клас. Скенерът може да обработва един файл или няколко файла едновременно (с посочени пътища), но сканирането на директории ще бъде разработено в бъдеще. Примерна команда на CLI скенера е:

```
python scanner.py --file suspicious.exe
```

Примерен изход от CLI скенера е:

```
[INFO] Scan result: suspicious.exe: Malicious | Critical Risk |
Probability: 95%
```

Действителен конзолен изход от CLI скенер, в който е показан пример за сканиране на 2 PE файла, един безобиден PE файл (python-3.8.10-amd64.exe) и един злонамерен PE файл (malicious_file_sample.exe). В изхода са предоставени категория риск, вероятностен резултат и запис в лога. Целият изход се записва в scan_log.txt, както е показано на Фиг. 3.6.

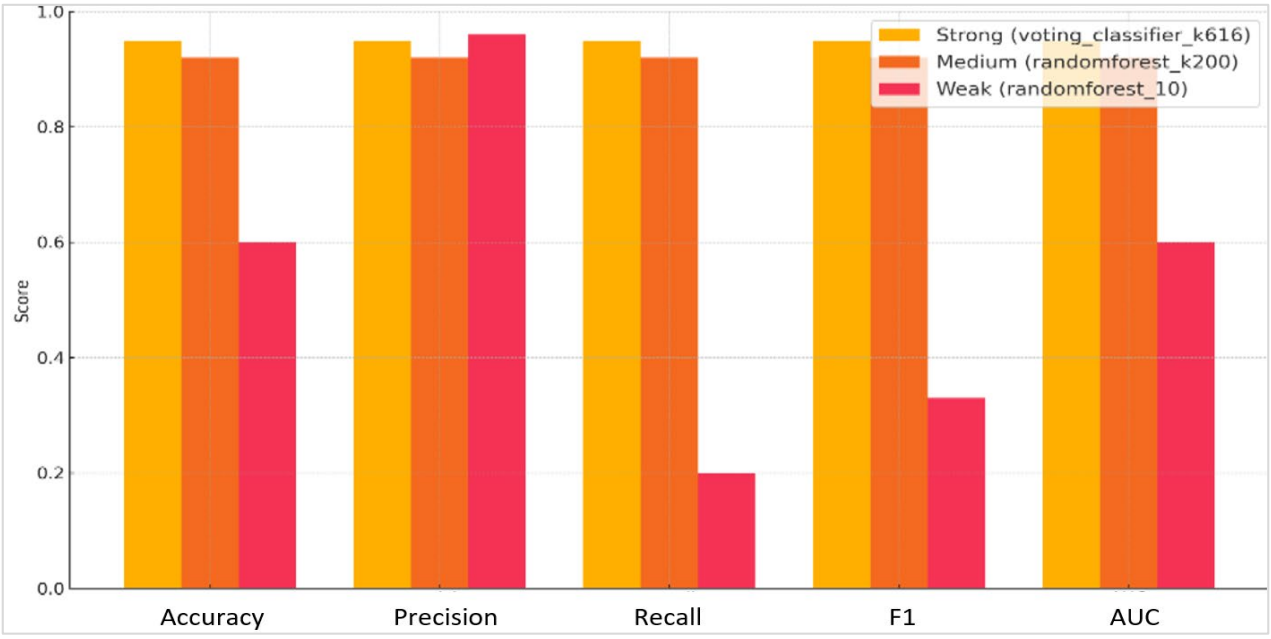


Фиг. 3.6. Действителен изход от конзолата от CLI скенер.

CLI скенерът идентифицира 5 нива на риск: 0: Доброкачествен; 1: Нисък риск; 2: Среден риск; 3: Висок риск; и 4: Критичен риск. Предложеният CLI скенер може да се използва от ИТ администратори на правителствени или корпоративни системи, които нямат достъп до интернет. CLI скенерът е съвместим с остарял хардуер със стари операционни системи. Този скенер може да се използва като част от автоматизация на скриптове или вътрешни защитни системи.

3.3.2. Визуализации и показатели

На Фиг. 3.8 е показана групирана стълбовидна диаграма, показваща три различни представителни модела от три различни класа на сила – силен (VotingClassifier), среден (Random Forest с k=200) и слаб (Random Forest с 10% обучаващ набор) – по отношение на пет показатели за оценка.

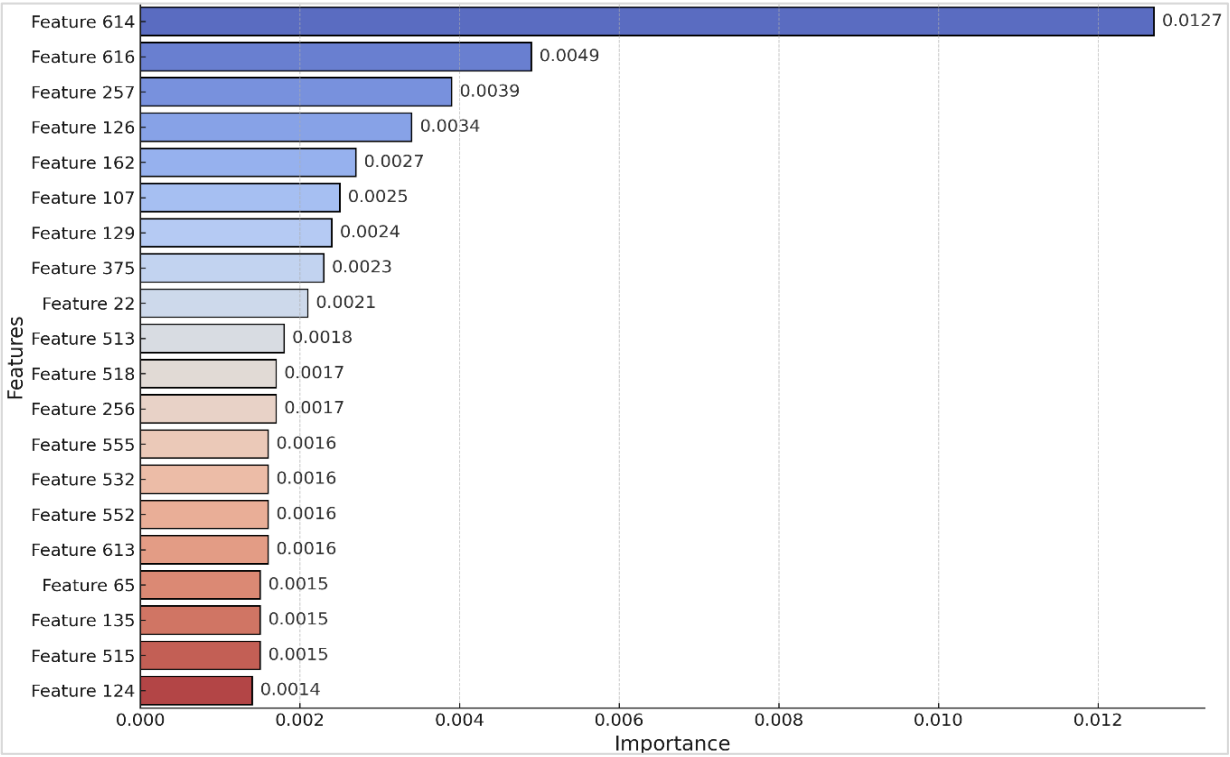


Фиг. 3.8. Сравняване на силни, средни и слаби модели по accuracy, precision, recall, F1 и AUC

ROC кривата, показана на Фиг. 3.9, е реализирана, за да се видят резултатите от силен модел (VotingClassifier), среден модел (Random Forest с k=200) и слаб модел (размер за обучение на Random Forest, 10%). AUC на VotingClassifier е най-висок (0.9424); слабият модел е малко по-добър от случайното предположение (0.5972). Разликата в AUC като цяло показва, че размерът за обучение е по-малко мощен от оптимизацията върху способността за класификация.

Фиг. 3.9. Сравнителни ROC криви

На Фиг. 3.11 са илюстрирани 20-те най-влиятелни характеристики, базирани на VotingClassifier.



Фиг. 3.11. 20-те най-влиятелни характеристики, базирани на VotingClassifier

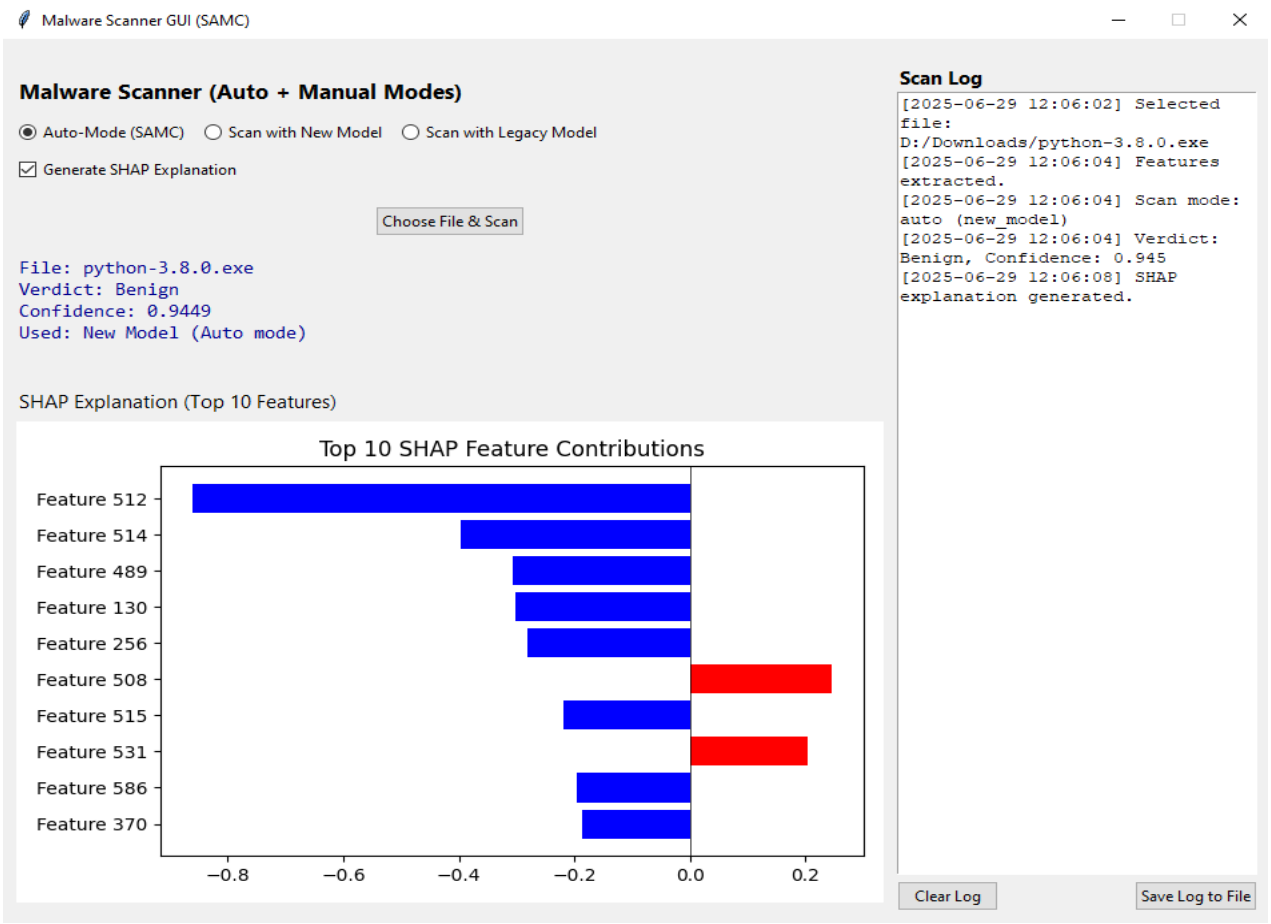
Характеристиките са подредени от най-голямо към най-малко влияние като идентификатор на характеристика (базирано на низходящи стойности), където характеристика 614 е най-влиятелната. Диаграмата е за обяснимост на модела и показва кои характеристики са отговорни за най-голямо влияние върху решенията.

3.4. Резултати от тестването на предложената самоосъзната класификация на зловреден софтуер чрез рутиране на модели на базата на система за доверие за избора и обяснимост на характеристиките

Този раздел орисвава проведените тестове чрез предложената рамка за откриване на зловреден софтуер, базирана на SAMC, измерваща производителността и оперативното поведение, съгласно описанието в Глава 2, раздел 2.4. Тук се представят не само проведените числени резултати, но и се анализира гъвкавостта на рамката, както и как разсъждения са обясними и по какъв начин рамката осигурява доверие и адаптивна промяна.

3.4.1. Дизайн и функции на графичния потребителски интерфейс

Дзайнът на потребителския интерфейс е разделен на 3 основни панела (Фиг. 3.12):



Фиг. 3.12. Примерен GUI интерфейс със SHAP обяснение за безвреден файл, сканиран чрез автоматичен режим, използвайки новия модел

Левият панел позволява на потребителя да избере режима за сканиране, да включва/изключва SHAP обясненията, да избере файл за сканиране и да види резултата. Десният панел позволява да се види лог на сканирането в реално време с времеви запис на съобщенията. Разделът с обяснения на SHAP представя хоризонтална стълбовидна диаграма, изобразяваща 10-те най-влиятелни характеристики, допринасящи за прогнозите, оцветени по знак (червено = положителен клас, синьо = отрицателен клас). Експериментален резултат от проведено сканиране на безвреден файл със SHAP обяснение е представен на Фиг. 3.12:

3.4.2. Време за изпълнение и латентност на рутирането

За да се определи възприеманата производителност в реално време на рамката SAMC, е направен бенчмарк на латентността на всички .exe примери, разположени в тестовата директория. Измерено е времето за изпълнение от точката, в която е започнато извличането на характеристики, до момента, в който е достигната прогнозата, идентифицирана от терминала, използвайки логиката за рутиране на SAMC (legacy model, new model или fallback). Сканирането на тези 100 файла може да се обобщи, както е показано в Таблица 3.6:

Таблица 3.6. Метрики и стойнсти от сканирането на 100 файла

Metric	Value (seconds)
Fastest	**0.0569**
Slowest	**0.5129**
Average	**0.2334**

Тези доказателства показват, че дори използването на логиката, базирана на доверието в прогнозата, е позволило класификация в почти реално време. Резервната логика е била използвана понякога за случаи, за които е установено, че няма несигурност относно резултата от прогнозата (напр. под прага), тъй като това е самоосъзнат дизайн, който отчита неяснотата/неизвестността или входните данни или моделите, които са предоставили оценки за доверие, оценени като ниски.

3.4.3. Сравнение на производителността: Рутиране срещу нерутиране

За да се оцени практическата ефективност на механизмите за рутиране, базирани на доверие, и резервните механизми, проведохме експеримент, в който оценихме четири различни стратегии за извод:

1. **Само Legacy Model** – използва VotingClassifier, обучен на EMBER 2018 и разчита единствено на исторически модели.
2. **Само New Model –** използва настроения и оптимизиран от Optuna класификатор XGBoost, обучен на EMBER 2024, който използва по-новото разпределение на данните.
3. **Комбинирано гласуване (без рутиране)** – просто наивен ансамбъл от двата модела с меко гласуване без никакви прагове за доверие.
4. **SAMC рутиране + резервен механизъм** – предложения интелигентен модел, който динамично избира модел въз основа на достоверността на прогнозата и който използва претеглена резервна логика, ако никой модел не премине прага.

За този експеримент бе създаден балансиран тестов набор от 50 известни проби от зловреден софтуер и 50 доброкачествени изпълними файла, като всеки файл премина през всичките четири канала за извод, след като беше статично анализиран в 616-измерен PE вектор на характеристиките. Резултатите са обобщени в Таблица 3.7:

Таблица 3.7. Сравнителна ефективност на различни стратегии за извод върху балансиран тестов набор (50 злонамерени и 50 доброкачествени файла).

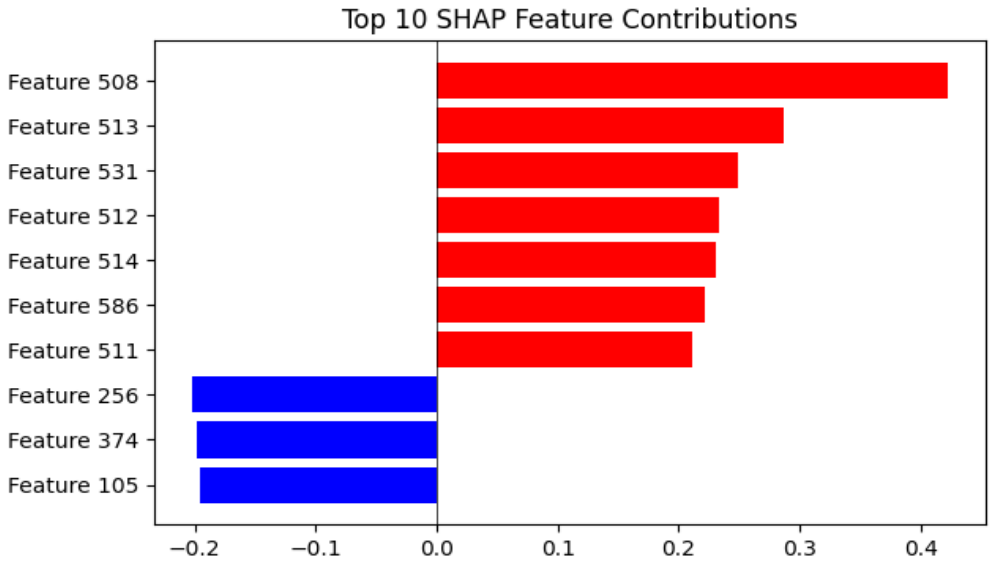
Mode	Accuracy	F1-Score	ROC-AUC	False Positives	False Negatives
Legacy Only (EMBER 2018)	0.6740	0.5000	0.9425	1	30
New Model Only (EMBER 2024)	0.8260	0.8421	0.8820	9	5
Combined Voting (No Routing)	0.8680	0.8608	0.9683	3	7
SAMC Routing + Fallback	0.9140	0.9036	0.9981	1	6

Тези резултати показват, че стратегията за рутиране SAMC е превъзхождала всички останали режими по всички показатели за оценка, постигайки F1-оценка над 0.9 и перфектна прецизност, но също така много висока попълнота и ROC-AUC. За разлика от традиционния модел, който показваше прекалено консервативно поведение при отчитане (без фалшиви положителни резултати, но с много ниска отзивчивост), SAMC постига среден вариант, тъй като или селективно се доверява на по-уверения модел, или прилага претеглен резервен метод. Това подкрепя заключението, че стратегията за избор на модел, съобразен с доверието, не само подобрява устойчивостта, но и помага за намаляване на фалшивите положителни и фалшиви отрицателни резултати при сравняване на производителността.

При изследване на ефективността на класификацията спрямо наивно ансамблово гласуване, подходът на SAMC беше по-добър, тъй като избягваше сляпото осредняване и вместо това се адаптираше, за да използва по-достоверния модел селективно за всяка извадка.

3.4.4. SHAP обяснение за зловреден файл

За да дадем още един пример за обяснимост и доверие, подготвихме локално SHAP обяснение за един от резултатите от злокачествени проби от новия модел. На Фиг. 3.13 са показани 10-те най-важни характеристики, допринесли за заключението „злонамерен“.



Фиг. 3.13. SHAP обяснение за злонамерен файл. Червените ленти показват положителен принос към злонамереното прогнозиране, докато сините ленти показват доброкачествени характеристики

SHAP визуално показва, че много от сегментите на ентропията и байтовата хистограма имат доминиращо значение за решението, което означава, че моделът силно обръща внимание на аномалиите в статистическите характеристики. Някои от същите характеристики също показват голямо отклонение в KS теста (напр. Характеристика 507), което подкрепя ротацията на адаптивното и обяснимото откриване.

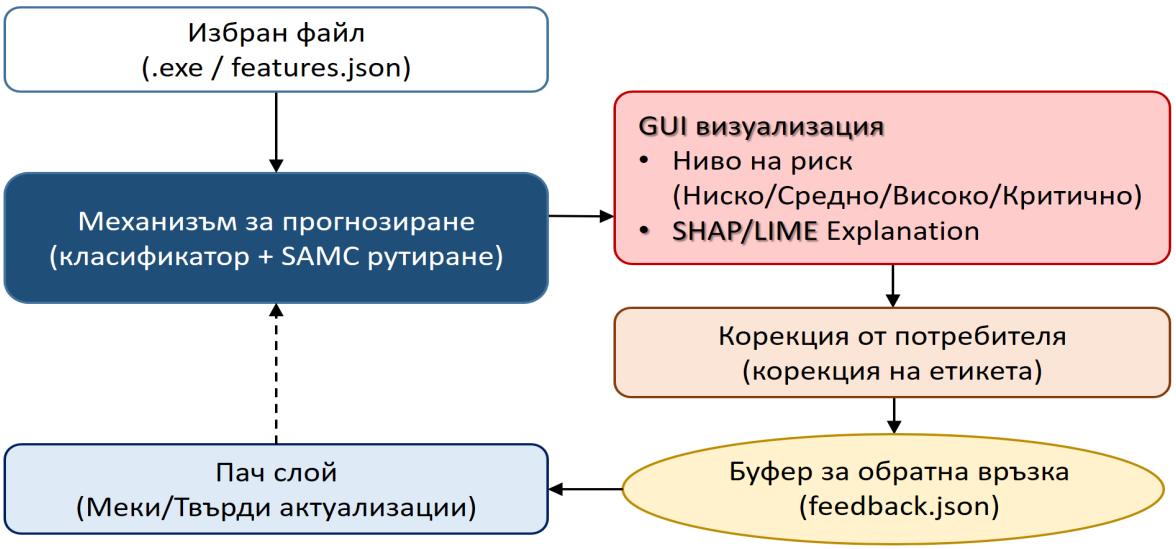
3.5. Резултати от разработеното и тествано приложение Shipka Guard на база на предложената адаптивна рамка, разчитаща на доверие за класификация на зловреден софтуер с корекции за обратна връзка

Графичният потребителски интерфейс (GUI) на разработеното приложение под името „Shipka Guard“ е изграден с помощта на рамката Streamlit. Това решение позволява бързо внедряване и предоставя достъп на анализаторите без инсталация или сложни конфигурации. В рамките на графичния потребителски интерфейс потребителите могат да качват PE файлове, да правят прогнози в реално време със свързани стойности на достоверност и да получават обясними (XAI) визуализации. Включени са както SHAP, така и LIME обяснения, с предоставен допълващ изглед; SHAP показва глобални приноси на характеристиките въз основа на архитектурата на модела, докато LIME показва локално интерпретируеми приближения въз основа на локалното решение.

Освен това преложението има опция, която може да се използва за ръчен избор на адаптивни версии в падащо меню, за да се сравняват и противопоставят паралелни адаптивни модели. Вграденият импорт/експорт на обратна връзка също позволява запазване или споделяне на обратната връзка от анализаторите, което допринася за възпроизводимостта на всеки експеримент и постига съвместен работен процес. Комбинацията от подробни регистрационни файлове и регулируема опция за прагове, с възможностите, прави Shipka Guard интерактивна изследователска и оперативна платформа.

В разаработената версия на приложението „Shipka Guard“ са интегрирани два XAI метода: (1) LIME локално обяснява всички модели и (2) SHAP глобално обяснява приноса на вградените характеристики в моделите, базирани на дървета (Adaptive XGBoost). Вградените визуализации се представят в графичния потребителски интерфейс, за да помогнат за разбирането и валидирането на решенията. Тези решения са интегрирана в предложената и реализирана архитектура на системата за машинно обучение с обратна връзка за класификация и обяснимост, паказана на Фиг. 3.14.

Всички системни събития (сканирания, записи за обратна връзка, преобучение) се регистрират в logs/scan.log. Освен това, рамката проследява времето за изпълнение (в ms) за всяка прогноза, за да гарантира, че системата ще работи в реално време на потребителския хардуер. Тази функция предоставя перспектива за осъществимостта и внедряването на проекта в корпоративни среди, включително резултати от сканиране, записи за обратна връзка, актуализации на корекции и събития за преобучение.



Фиг. 3.14. Архитектура на системата за машинно обучение с обратна връзка за класификация и обяснимост (XAI)

3.5.1. Оценка на разработеното приложение Shipka Guard, на база на предложената рамка

Оценката на приложимостта на разработеното приложение Shipka Guard, на база на предложената рамка, използва два публично достъпни набора от данни за бенчмаркове: EMBER 2018, съдържащ над 1 милион PE проби, характеризиращи се с 616 статични характеристики (т.е. хистограми, ентропия, низови статистики) и използван като базов „наследен“ модел за изследването; и EMBER 2024 с приблизително 960 000 проби. EMBER 2024 съдържа съвременни семейства злонамерен софтуер и е хармонизиран в същото 616-измерно пространство от характеристики за сравнение с набора от данни EMBER 2018. Освен това се използва трети, генериран от потребителя набор от данни. Рамката е структурирана така, че да интегрира генерирани от потребителя данни по лесен начин, за да се хармонизира в същото 616-измерно пространство от характеристики.

За целите на честност и възпроизводимост на експериментите, от EMBER2018 и EMBER2024 беше създаден балансиран набор от 100 000 проби (50 000 доброкачествени, 50 000 злонамерени). Първоначално базовите модели и методът за автоматично рутиране SAMC бяха сравнени при балансираната оценка, както е показано в Таблица 3.8.

Таблица 3.8. Производителност на Legacy, Modern и SAMC Auto (T=0.85)

Режим	Accuracy	Precision	Recall	F1	ROC-AUC	FN	FP
Force Legacy	0.92409	0.9540	0.8906	0.9213	0.9621	5472	2148
Force Modern	0.96641	0.9793	0.9529	0.9660	0.9955	2353	1006
Auto (SAMC) [T=0.85]	0.95917	0.9852	0.9324	0.9580	0.9942	3381	702

Разгледания ефект от леката адаптация на Patch спрямо пълното преобучение, както е показано в Таблица 3.9.

Таблица 3.9. Аблационно проучване, сравняващо изходните данни Modern, Patch с 500 проби за обратна връзка и пълно преобучение с 10 000 проби за обратна връзка

Режим	Accuracy	Precision	Recall	F1	ROC-AUC	FN	FP
Base (Modern)	0.96641	0.9793	0.9529	0.9660	0.9955	2353	1006
Patch (500 fb)	0.96727	0.9799	0.9541	0.9668	0.9918	2294	979
Full Retrain (10k)	0.94719	0.9672	0.9257	0.9460	0.9906	3713	1568

Слоят Patch, обучен с 500 проби за обратна връзка, показва забележими подобрения: както FN (-59), така и FP (-27) намаляват, а резултатът F1 леко се увеличава. По този начин, дори малък и евентуално небалансиран буфер за обратна връзка може да се използва за прилагане на последователни, бързи корекции. За разлика от това, пълното преобучение с 10 000 проби за обратна връзка дава по-лош резултат. Въпреки че изглежда нелогично, обяснението се корени в пристрастия и дисбаланс в пула за обратна връзка: ние умножаваме проблемните проби и увеличаваме грешките, вместо да ги поправяме. Това наистина предоставя критичен урок: качеството и разнообразието са също толкова важни, колкото количеството на обратната връзка.

SAMC Auto, използващ различни прагове на доверие, е показан в Таблица 3.10.

Таблица 1.10. Метрики при различните прагове на доверие.

T	Precision	Recall	%Uncertain
0.80	0.9854	0.9310	6.5%
0.85	0.9852	0.9324	8.2%
0.90	0.9851	0.9341	10.6%

По-ниските прагове (T=0.80) водят до по-малко въздържания, но по-голям риск от погрешни класификации. Използването на по-високи прагове (T=0.90) увеличава броя на въздържанията, което също подобрява надеждността, но е съпроводено с намаляване на пропускателната способност. Това допълнително подкрепя идеята, че T е регулируем бутон за безопасност.

Времето за изпълнение на файл е измерено директно от системните лог файлове за всеки модел и резултатите са дадени в Таблица 3.11.

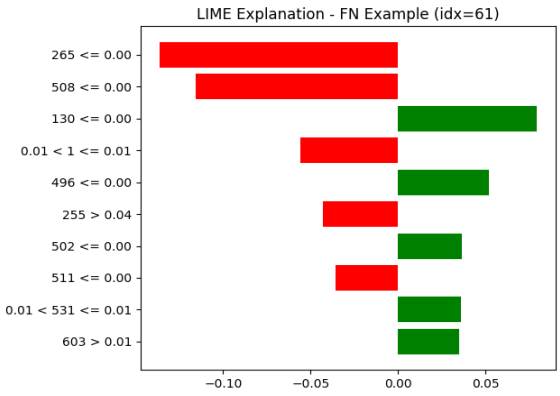
Таблица 3.11. Латентност на изпълнението, нормализирана до 100 проби на модел

Режим	n_samples	Mean (ms)	Std (ms)	p50	p90	p99
Modern	100	47.6	24.5	33.1	83.6	91.3
Legacy	100	131.4	0.0	131.4	131.4	131.4
Adaptive	100	109.7	29.5	109.7	133.3	138.6

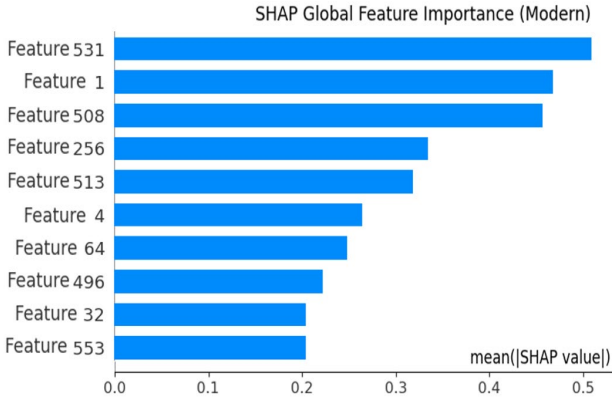
Моделът Modern постига средно време за извод под 50 ms, което е доста под целевата граница от 90 ms/проба. Legacy и Adaptive са по-бавни (~110–130 ms), но все още в рамките на реалното време. Тези резултати потвърждават възможността за внедряване на системата дори с активиран SHAP/LIME.

Три примерни казуса, илюстриращи динамиката на обратната връзка и обясненията:

- **Фалшиво отрицателен резултат, коригиран чрез Patch.** На Фиг. 3.15 (LIME) е показаня злонамерена проба, погрешно класифицирана като доброкачествена. След корекция с 500 семпли за обратна връзка, решението беше коригирано, намалявайки FN.
- **Фалшиво положителен резултат, коригиран чрез обратна връзка.** Доброкачествена проба, погрешно класифицирана като злонамерена, беше добавена към буфера за обратна връзка. Моделът Patch беше коригиран и фалшиво положителният резултат беше елиминиран.
- **Несигурен случай, обяснен чрез LIME и SHAP.** При T=0.90, SAMC се въздържа. LIME показа противоречиви локални доказателства, докато глобалното значение на SHAP (Фиг. 3.16) обясни влиянията на доминиращите характеристики.



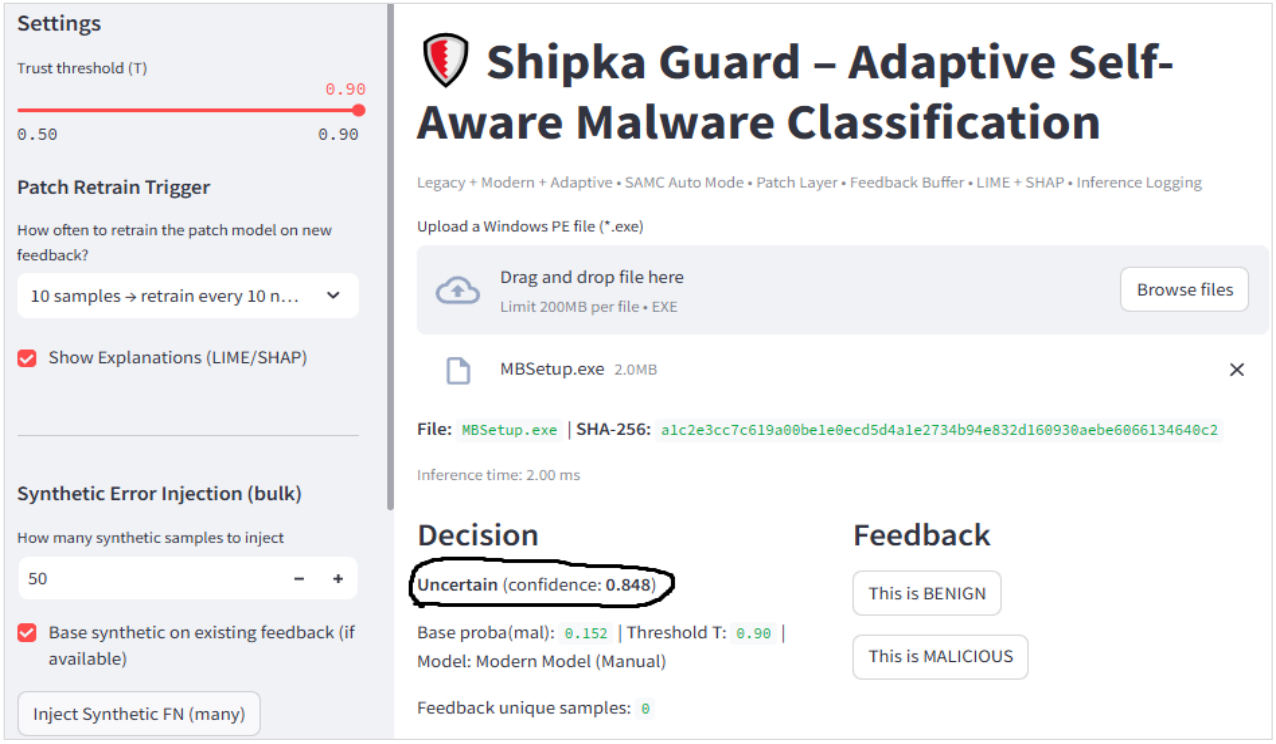
Фиг. 3.15. Обяснението на LIME за FN е коригирано от Patch



Фиг. 3.16. Важност на глобалните характеристики на SHAP за Modern модела

GUI, показан на Фиг. 3.17, показа въздържанието на анализатора, който може да предостави коригираща обратна връзка.

Оценката потвърждава ефективността и ограниченията на предложената рамка. Старите модели са остарели, докато съвременните модели се представят добре, но произвеждат повече FP. SAMC Auto балансира рисковете чрез динамично рутиране. Слоят Patch осигурява лесно и леко почистване на грешки с малки обработени пулове, а пълното преобучение не дава превантивно предимство, освен ако няма подходящо количество обратна връзка.



Фиг. 3.17. Екранна снимка на потребителския интерфейс с „Несигурно решение“ (T=0.90)

Прагът на доверие предлага настройваем баланс между попълнота и въздържания. Латентността остава в рамките на реалното време, докато казусите показват как модулът за обяснение (LIME/SHAP) и несигурните въздържания впоследствие са увеличили доверието на анализаторите или поне са валидирали резултатите. Следователно, ние обосноваваме процеса SAMC като ефективен и надежден процес за адаптиране на класификацията на зловредния софтуер.

3.5.2. Дискусии и изводи

Shipka Guard интегрира 4 режима на сканиране, инжектиране на синтетични грешки, експорт/импорт на обратна връзка, базиран на JSON, и мащабируем анализ. Това прави рамката Shipka Guard не само по-адаптивна, но и по-подходяща за реални условия на внедряване. Рамката Shipka Guard е базирана на интерактивен графичен потребителски интерфейс, който намалява бариерата за неспециализирани потребители да експериментират с класификацията на зловреден софтуер. Буферът за обратна връзка позволява на потребителя съвместно да коригира модела, което му позволява да експортира/импортира файлове с обратна връзка. Интеграцията на обяснимостта допринася за доверието в решенията, което се е увеличило чрез повече информация за функциите от локална и глобална гледна точка. Регистрирането на мащабируемостта отразява адаптивността на системата, което не възпрепятства работните процеси в реално време. Всички тези аспекти показват, че адаптивното откриване на зловреден софтуер може да бъде практично и прозрачно.

Рамката предлага няколко нови приноса. Първият от тях е механизъм за рутиране, съобразен с доверието, който използва множество поколения модели, включително резервен механизъм. Следващият е буфер за обратна връзка и слой за корекции за лека адаптация от потребителски корекции и синтетични инжекции. Реализацията му се осъществява чрез интерактивен графичен потребителски интерфейс Streamlit, който предоставя четири режима на сканиране, споделяне на обратна връзка на базата на JSON и локални/глобални обяснения (LIME+SHAP). Той също така осигурява мащабируемо регистриране, което представя изводи в реално време (<90ms) на потребителски хардуер.

Комбинацията от адаптивност, интерпретируемост и мащабируемост води до справяне с предизвикателствата на непрекъснатото разработване на практични системи за откриване на зловреден софтуер от следващо поколение, които постоянно остават „верни“ за дадените модели на заплахи, вместо да се променят постоянно. Включването на избор на версия и интерфейс за обратна връзка като част от приложението позволява възпроизводимост и съвместна използваемост, което прави Shipka Guard не само практичен инструмент за защита от зловреден софтуер, но и инструмент за съвместни изследвания или оценки.

ЗАКЛЮЧЕНИЕ – РЕЗЮМЕ НА ПОЛУЧЕНИТЕ РЕЗУЛТАТИ

Зловредният софтуер (malware) може да приеме много и различни форми, като вируси, червеи, троянски коне и ransomware, и може да причини значителни вреди на хора, организации и дори на цели държави. Зловредният софтуер остава една от най-сериозните заплахи за информационната сигурност. С еволюцията на дигиталните технологии се променят и методите за извършване на атаки, което налага разработване на нови алгоритми за ефективно и навременно откриване на зловредно поведение. Традиционните решения, базирани предимно на сигнатури, вече не са достатъчни за справяне със съвременните заплахи.

Отчитайки необходимостта от разработването на ефективни средства за противодействие на зловредния софтуер, определи и основната цел на настоящия дисертационен труд, свързана с Изследване и анализ на възможностите за откриване на злонамерен софтуер чрез средствата на машинно обучение. За де се реализират експериментите безопасно в контролирана среда е предложен математически модел за избор на подходяща виртуална машина. При наличието на безопасна в контролирана среда са проведени експерименти за установяване ефективността на различни алгоритми на машинното обучение и е направен анализ на тяхното представяне за целите на откриване на зловреден софтуер. На база на установените констатации е предложен подобрен подход за статичен анализ за откриване на зловреден софтуер чрез оптимизиране извличането на характеристики и комбинирайки различни алгоритми за машинно обучение. В допълнение на тези усилия е предложена рамка за статична

класификация на зловреден софтуер, използваща оптимизация на функции и ансамблово обучение. Резултатите от проведеното тестване показват, че анализът на фалшиво положителните резултати за ансамбъла е значително по-нисък от този на отделните модели. За подобряване на класификация на зловреден софтуер е предложена самоосъзната рамка, интегрираща рутиране на модели на базата на система за доверие за избора и обяснимост на характеристиките. Логиката на рутиране увеличава мощността на ансамбъла с решения, базирани на доверие, и предоставя гъвкав механизъм, полезен както за минали, така и за съвременни характеристики на зловредния софтуер. За прецизиране на класификация на зловредния софтуер е предложена адаптивна рамка, съобразена с доверието, позволяваща класификация на зловреден софтуер с възможност за корекции чрез обратна връзка. Буферът за обратна връзка позволява на потребителя съвместно да коригира модела, което му позволява да експортира/импортира файлове с обратна връзка. Интеграцията на обяснимостта допринася за доверието в решенията, което се е увеличило чрез повече информация за функциите от локална и глобална гледна точка. Тази адаптивна рамка е реализирана в разработена демо версия на софтуерно приложение под името „Shipka Guard“. Тя интегрира механизъм за рутиране, съобразен с доверието, който използва множество поколения модели, включително резервен механизъм. В нея се използва буфер за обратна връзка и слой за корекции за лека адаптация от потребителски корекции и синтетични инжекции. Благодарение на комбинацията от адаптивност, интерпретируемост и мащабируемост се постига решават предизвикателства, свързани с непрекъснатото разработване на практични системи за откриване на зловреден софтуер от следващо поколение, които постоянно остават „верни“ за дадените модели на заплахи, вместо да се променят постоянно.

Чрез проведените многобройни експерименти с различни видове данни е установена практическата приложимост както на предложените хибридни алгоритми така и на разработеното приложение под името „Shipka Guard“.

Като бъдещо развитие на изследванията в дисертационния труд се планира изследвания, свързани с едновременно и многокласово идентифициране на зловреден софтуер.

Получените резултати по темата на дисертационното изследване са докладвани на 3 международни конференции. Представените резултати са отразени в общо 4 научни публикации в издания, които са реферирани и индексирани в световноизвестни бази данни с научна информация – Scopus.

ПРИНОСИ

Получените резултати, описани в настоящия дисертационен труд, могат да се обобщят в следните научни и научно-приложни приноси:

1. Предложени са два матаматически модела, чрез които може да се направи избор на софтуер за подходяща виртуална машина за целите на експерименталното тестване за откриване на зловреден софтуер.
2. Предложен е подобрен подход за статичен анализ за откриване на зловреден софтуер чрез оптимизиране извличането на характеристики чрез комбиниране на различни алгоритми за машинно обучение. Проведените тестове с предложените хибридни алгоритми показват по-добра производителност.
3. Предложена е рамка за статична класификация на зловреден софтуер, която използва оптимизация на функции и ансамблово обучение. Резултатите показват, че анализът на фалшиво положителните резултати за ансамбъла е значително по-нисък от този на отделните модели.
4. Предложена е самоосъзната класификация на зловреден софтуер чрез рутиране на модели на базата на система за доверие за избора и обяснимост на характеристиките. Логиката на рутиране увеличава мощността на ансамбъла с решения, базирани на доверие, и предоставя гъвкав механизъм, полезен както за минали, така и за съвременни характеристики на зловредния софтуер.
5. Предложена е адаптивна рамка, съобразена с доверието, за класификация на зловреден софтуер с корекции за обратна връзка. Тази рамка е едновременно адаптивна и устойчива, тъй като включва самоосъзнат класификатор на модели, който използва адаптивна логика за автоматичен избор между традиционни и съвременни слоеве на моделите чрез измерване на надеждността на прогнозирането. Интеграцията на обяснимостта допринася за доверието в решенията, което се е увеличило чрез повече информация за функциите от локална и глобална гледна точка.

СПИСЪК НА ПУБЛИКАЦИИТЕ ПО ДИСЕРТАЦИОННИЯ ТРУД

1. **Barzev**, I., Borissova, D.: An improved static analysis approach for malware detection by optimizing feature extraction combining different ML algorithms. In: Bennour, A., Bouridane, A., Almaadeed, S., Bouaziz, B., Edirisinghe, E. (eds) Intelligent Systems and Pattern Recognition. ISPR 2024. Communications in Computer and Information Science, vol. 2304, pp. 102–115, 2025, Springer, Cham. https://doi.org/10.1007/978-3-031-82153-0_8. Print ISBN 978-3-031-82152-3, **SJR Q4=0.182**
2. **Barzev**, I., Borissova, D.: Performance analysis of LSTM, SVM, CNN and CNN-LSTM algorithms for malware detection in IoT dataset. WSEAS TRANSACTIONS on COMPUTER RESEARCH, vol. 13, 288–296, 2025, http://dx.doi.org/10.37394/232018.2025.13.27. E-ISSN: 2415-1521, **SJR Q4=0.140**
3. **Barzev, I.**, Borissova, D., Buhtiyarov, N.: Comparison of different binary classification algorithms for malware detection. In: Rocha, Á., Ferrás, C., Hochstetter Diez, J., Diéguez Rebolledo, M. (eds) Information Technology and Systems. ICITS 2024. Lecture Notes in Networks and Systems, vol. 932, pp. 369–378, 2024, Springer, Cham. https://doi.org/10.1007/978-3-031-54235-0_33. Print ISBN 978-3-031-54234-3, Online ISBN 978-3-031-54235-0, **SJR Q4 =0.17.**
4. Borissova, D., **Barzev, I.**, Yoshinov, R., Kotseva, M.: Group decision-making models for selection of virtual machine software for malware detection purposes. In: Proc. of 12th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 2023, https://doi.org/10.1109/MECO58584.2023.10155084. ISBN 979-8-3503-2291-0.

Публикации, приети за петат:

- Barzev, I., Borissova, D.: Trust-Aware Adaptive Framework for Malware Classification with Feedback Patching. 9th International Conference on Information Technology & Systems, 16-18 February 2026
- Barzev, I., Borissova, D.: A scalable Python Framework for static malware classification using feature optimization and ensemble Learning. International Conference on Advanced Research in Technologies, Information, Innovation, and Sustainability (ARTIIS 2025). 21-23 October 2025.
- Barzev, I., Borissova, D.: Self-aware malware classification via confidence-guided model routing and explainable feature attribution. 5th International Conference on Intelligent Systems and Pattern Recognition, 25-27 September 2025

БИБЛИОГРАФИЯ

1. Ozkan-Okay, M., Akin, E., Aslan, O., Kosunalp, S., Iliev, T., Stoyanov, I., Beloev, I.: A comprehensive survey: Evaluating the efficiency of artificial intelligence and machine learning techniques on cyber security solutions. IEEE Access, vol. 12, pp. 12229-12256, 2024, https://doi.org/10.1109/ACCESS.2024.3355547.
2. Barzev, I., Borissova, D.: Performance analysis of LSTM, SVM, CNN and CNN-LSTM algorithms for malware detection in IoT dataset. WSEAS TRANSACTIONS on COMPUTER RESEARCH, vol. 13, 288-296, 2025, http://dx.doi.org/10.37394/232018.2025.13.27.
3. Barzev, I., Borissova, D., Buhtiyarov, N.: Comparison of different binary classification algorithms for malware detection. In: Rocha, Á., Ferrás, C., Hochstetter Diez, J., Diéguez Rebolledo, M. (eds) Information Technology and Systems. Lecture Notes in Networks and Systems, vol. 932, pp. 369-378, 2024, https://doi.org/10.1007/978-3-031-54235-0_33.
4. Praveen, E. Kumar, S. Priyanka, A comprehensive survey on hardware-assisted malware analysis and primitive techniques, Computer Networks, vol. 235, 109967, 2023, https://doi.org/10.1016/j.comnet.2023.109967.
5. Borissova, D., Barzev, I., Yoshinov, R., Kotseva, M.: Group decision-making models for selection of virtual machine software for malware detection purposes. In: Proc. of 12th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 2023, https://doi.org/10.1109/MECO58584.2023.10155084.
6. Barzev, I., Borissova, D.: An improved static analysis approach for malware detection by optimizing feature extraction combining different ML algorithms. In: Bennour, A., Bouridane, A., Almaadeed, S., Bouaziz, B., Edirisinghe, E. (eds) Intelligent Systems and Pattern Recognition. ISPR 2024. Communications in Computer and Information Science, vol. 2304, pp. 102–115, 2025, https://doi.org/10.1007/978-3-031-82153-0_8.
7. Ho, T.K., Hull, J.J., Srihari, S.N.: Decision combination in multiple classifier systems. IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 16(1), pp. 66-75, 1994, https://doi.org/10.1109/34.273716.
8. Kittler, J., Hatef, M., Duin, R.P.W., Matas, J.: On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20(3), pp. 226-239, 1998, https://doi.org/10.1109/34.667881.
9. Thai, H.-T.: Machine learning for structural engineering: A state-of-the-art review. Structures, vol. 38, pp. 448-491, 2022, https://doi.org/10.1016/j.istruc.2022.02.003.
10. Hancock, J.T., Khoshgoftaar, T.M. CatBoost for big data: an interdisciplinary review. Journal of Big Data, vol. 7, 94, 2020, https://doi.org/10.1186/s40537-020-00369-8.