DOI 10.36622/1682-7813.2025.28.1.010
УДК 004.853

НЕЙРОСЕТЕВАЯ СИСТЕМА ПРОТИВОДЕЙСТВИЯ КИБЕРАТАКАМ НА ОСНОВЕ МОДЕЛИ LLAMA 3.2

Н.П. Жуков, В.И. Белоножкин, Г.А. Остапенко, А.П. Васильченко, А.А. Остапенко, Е.Ю. Чапурин

В статье представлена самообучающаяся нейросетевая система противодействия кибератакам, основанная на модели LLAMA 3.2. Система объединяет алгоритмы дообучения, backend-сервер на FastAPI, frontend-приложение на React и базу данных TinyDB. Разработан алгоритм дообучения модели, обеспечивающий её адаптацию к новым угрозам и минимизацию затрат на переобучение. Архитектура системы охватывает взаимодействие её компонентов, включая обработку данных, инференс и автоматизированное управление процессами через API. Практическая ценность работы заключается в создании инструмента, способного обеспечивать актуальную защиту информационных систем и адаптироваться к динамично меняющимся сценариям киберугроз. Разработанные архитектурные решения и алгоритмы могут быть применены для различных задач адаптивного управления, связанных с обработкой больших данных и автоматизацией анализа угроз, а также способствовать повышению эффективности технологий для защиты информационной среды.

Ключевые слова: нейросеть, система противодействия, уязвимости, защита информационной среды, LLAMA 3.2.

Введение

Расширение применения технологий цифровизации приводит к росту количества и сложности киберугроз [1]. Современные атаки, такие как эксплойты нулевого дня, фишинг и вредоносное ПО, обходят традиционные методы защиты, становясь всё более сложными и труднообнаружимыми. Это требует разработки технологий, например нейросетевых моделей, способных дообучаться для эффективного противодействия новым угрозам в меняющейся киберсреде [2].

Ключевые преимущества нейросетевых технологий — это эффективная обработка больших объёмов данных в реальном времени, высокая гибкость для оперативного реагирования на угрозы и возможность дообучения для борьбы с новыми сценариями атак. Эти технологии также повышают точность реагирования и минимизируют последствия атак [3].

Модель LLaMA 3.2 (Large Language Model Meta AI) — мощная языковая модель от Meta AI на основе трансформеров, эффективно обрабатывающая текстовую и визуальную информацию. Мультимодальные возможности делают её полезной для кибербезопасности, автоматизируя анализ угроз и создание инструкций для реагирования на атаки [3, 4].

Предотвращение атак в данном контексте включает в себя разработку рекомендаций и регламентов, которые определяют последовательность действий для оперативного реагирования на инциденты. Также важным аспектом является разработка мер по ликвидации последствий атак, минимизации ущерба и восстановлению нормальной работы систем. В конечном итоге это обеспечивает системный подход к управлению киберугрозами, что способствует повышению уровня защиты и устойчивости инфраструктуры [2, 3, 6].

Для эффективного использования модели LLaMA 3.2 с целью повышения защиты автоматизированных информационных систем (АИС) от современных кибератак было необходимо развить функциональные и сервисные возможности специализированной нейросети [6]. В рамках этого подхода решались следующие ключевые задачи:

©Жуков Н.П., Белоножкин В.И., Остапенко Г.А., Васильченко А.П., Остапенко А.А., Чапурин Е.Ю., 2025

1) разработка алгоритма дообучения, который позволяет модели не только адаптироваться к новым сценариям угроз, но также сохранять и использовать накопленные знания для повышения точности и устойчивости в борьбе с киберугрозами. Это гарантирует, что система будет эффективно реагировать на изменения в характеристиках атак и сохранять способность к обучению на новых данных без потери старых знаний.

2) проектирование и реализация архитектуры системы, обеспечивающей интеграцию всех ключевых компонентов и масштабируемость. Это включает в себя как создание инфраструктуры для обработки и хранения данных, так и обеспечение возможности расширения системы для обработки больших объёмов информации и поддержания высоких требований к производительности при росте нагрузки.

Таким образом, использование модели LLaMA 3.2 в системе направлено не только на повышение её эффективности в борьбе с киберугрозами, но и на создание гибкой и адаптивной структуры, способной развиваться и реагировать на постоянно меняющиеся вызовы в сфере кибербезопасности.

Алгоритм дообучения модели

Разработанный алгоритм, обеспечивающий адаптацию нейросети и минимизирующий необходимость ее полного переобучения, включает следующие шаги, указанные на рис. 1:

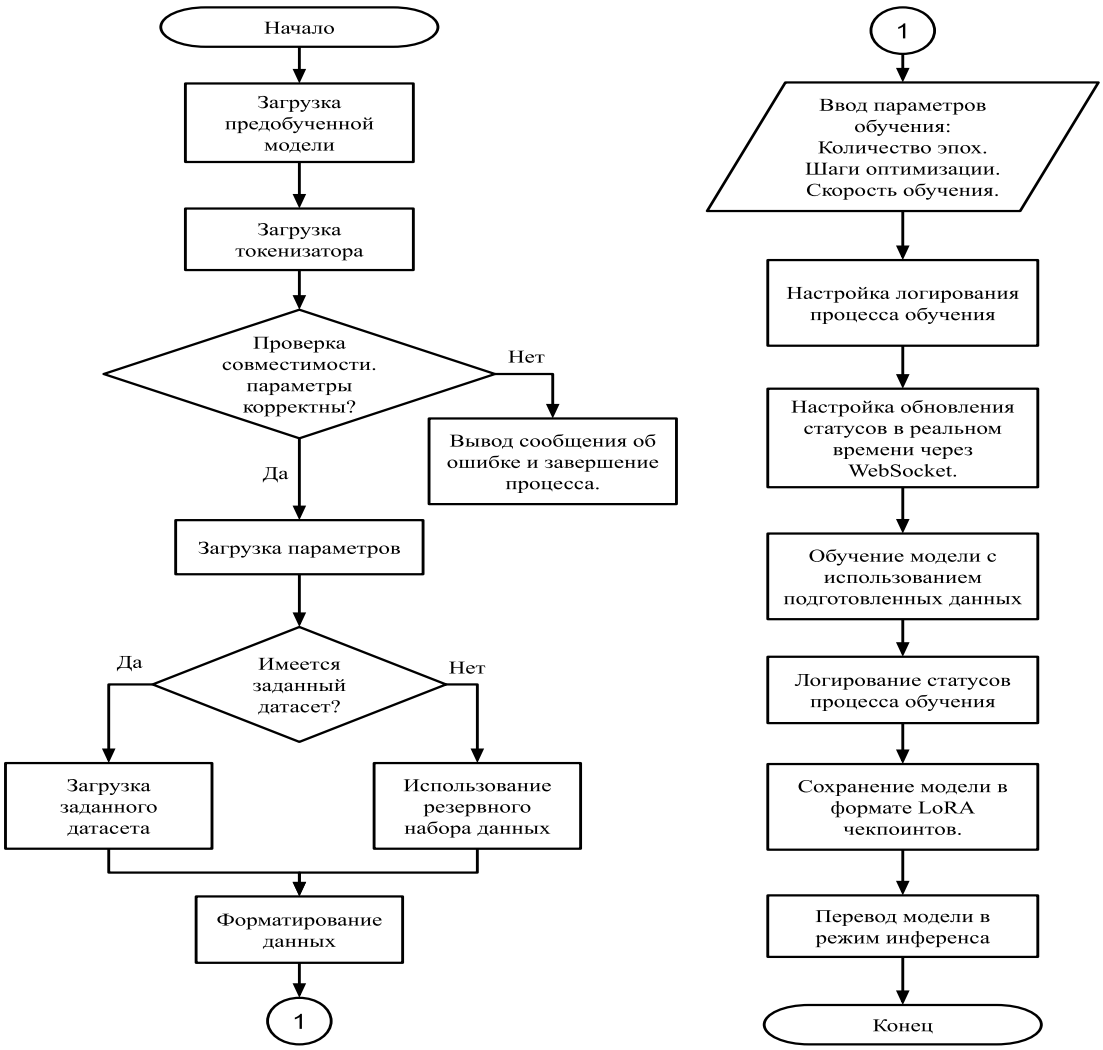

Рис.1. Алгоритм дообучения нейросети

1) перед началом обучения загружается предобученная модель и токенизатор, который служит для разбиения текста на составляющие элементы (токены), такие как слова, фразы или символы. На этом этапе проверяется их совместимость и корректность параметров. Для обеспечения масштабируемости и производительности модели применяется технология LoRA (Low-Rank Adaptation), которая оптимизирует процесс обучения, изменяя только ключевые параметры без ущерба для уже накопленных знаний [7].

2) затем загружаются обучающие датасеты, предоставляемые администратором через интерфейс прикладного программирования (API). Если заданный датасет отсутствует, используется резервный набор данных. В процессе подготовки данные проходят автоматическое форматирование для приведения их к унифицированному виду, необходимому для обработки моделью. Это достигается за счёт использования функции преобразования входных данных, которая формирует текстовые последовательности с учётом структуры модели [8].

3) обучение модели выполняется с использованием заранее определённых параметров, таких как количество эпох, шаги оптимизации и метод адаптивного управления скоростью обучения в процессе обучения ведётся детализированное логирование статуса. Обновления статуса доступны администраторам в режиме реального времени через протокол связи поверх tcp-соединения для обмена сообщениями между браузером и веб-сервером (WebSocket) [9].

4) по завершении обучения модель сохраняется в формате чекпоинтов LoRA, которые позволяют восстановить её состояние на любом этапе. Администраторы могут управлять этими чекпоинтами через API, включая их загрузку, удаление и скачивание.

5) после завершения обучения модель тестируется для проверки её работоспособности. Затем она переводится в режим инференса, который обеспечивает возможность обработки запросов от пользователей.

Этот алгоритм обеспечивает высокую адаптивность модели, позволяя оперативно реагировать на изменения в ландшафте киберугроз. Реализация через API обеспечивает удобство управления процессом обучения и интеграцию в существующую инфраструктуру. Такой подход позволяет эффективно использовать ресурсы и минимизировать время на внедрение обновлений.

Архитектура разработанной системы

Архитектура разработанной системы реализует современную клиент-серверную технологию и включает три основных компонента, которые изображены на рис. 2:

– пользовательское приложение (frontend);

– серверное приложение (backend);

– базу данных (БД).

Такое распределение компонентов позволяет организовать удобное и эффективное взаимодействие между пользователями и системой, обеспечивая интуитивно понятный интерфейс и высокую производительность.

Каждый из компонентов системы выполняет свою роль в общем процессе противодействия кибератакам:

1. Пользовательское приложение (frontend) предоставляет интерфейс для взаимодействия пользователя с системой. С его помощью можно инициировать задачи, такие как обучение модели, проведение инференса, анализ данных, а также получать результаты работы системы.

2. Серверное приложение (backend) выполняет обработку запросов от клиентской части, управляет взаимодействием с нейросетевой моделью и базой данных, а также обеспечивает выполнение ключевых функций, таких как загрузка и дообучение модели.

3. База данных (БД) отвечает за хранение обучающих данных, параметров моделей, логов работы системы и других служебных данных, необходимых для её функционирования.

Такая архитектура обеспечивает гибкость и масштабируемость системы, позволяя легко интегрировать

дополнительные модули или расширять функциональные возможности.

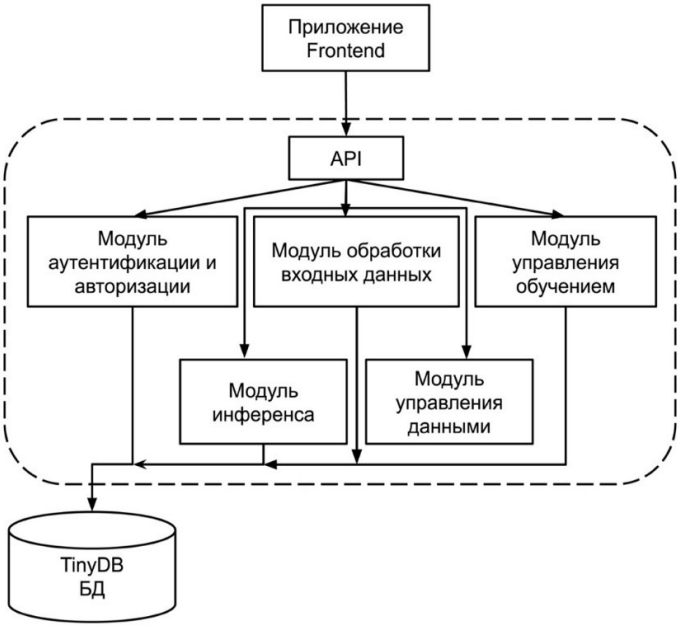

Рис.2. Архитектура нейросетевой системы противодействия кибератакам

Frontend-приложение реализовано с использованием фреймворка React и инструментов сборки Vite. Эти технологии выбраны для обеспечения высокой производительности, удобства разработки и масштабируемости интерфейса.

React — это JavaScript-библиотека с открытым исходным кодом для создания интерактивных пользовательских интерфейсов. Её основное преимущество заключается в компонентном подходе, который позволяет разбивать интерфейс на независимые модули, упрощая поддержку, тестирование и масштабирование приложений. React активно используется для создания динамичных приложений благодаря следующим характеристикам:

– виртуальный DOM – обновление только изменённых элементов пользовательского интерфейса, что обеспечивает высокую производительность;

– компонентная архитектура, которая позволяет повторно использовать код, снижая сложность разработки;

– универсальность, позволяющая создавать веб- и мобильные приложения (с использованием React Native);

– большое количество библиотек и инструментов.

Vite (с французского «быстрый») — современный инструмент сборки веб-приложений, использующий подход, основанный на ES-модулях, что значительно ускоряет процесс разработки. Ключевые преимущества Vite:

– загрузка только тех файлов, которые необходимы, что ускоряет процесс старта;

– горячая перезагрузка (Hot Module Replacement) – изменения в коде моментально отображаются в браузере, упрощая отладку и тестирование;

– интеграция с другими популярными фреймворками (React, Vue, Svelte, TypeScript и другие технологии);

– автоматическая минимизация и оптимизация ресурсов.

Фронтенд-приложение, реализованное на базе React и Vite, позволяет пользователям легко управлять процессами обучения и инференса, загружать и форматировать данные. Это делает систему не только функциональной, но и удобной в эксплуатации.

Backend-сервер играет ключевую роль в работе всей системы, обеспечивая её стабильность, производительность и масштабируемость. Для его реализации был выбран фреймворк FastAPI, который

зарекомендовал себя как один из самых быстрых и эффективных инструментов для создания API. FastAPI поддерживает асинхронные операции, что позволяет обрабатывать множество запросов одновременно, обеспечивая низкие задержки и высокую пропускную способность. Это особенно важно в условиях современных требований к быстродействию и масштабируемости в сфере кибербезопасности.

Backend-приложение выполняет несколько ключевых функций, которые обеспечивают полноценную работу системы:

1) инференс и дообучение модели. Сервер управляет процессами загрузки предобученной модели, её адаптации к новым данным и выполнением запросов инференса. Это позволяет системе оперативно обрабатывать запросы пользователей и адаптироваться к изменяющимся условиям, обеспечивая эффективное взаимодействие с моделью.

2) управление данными. Сервер поддерживает загрузку, форматирование и хранение обучающих наборов данных, а также управление чекпоинтами модели, что обеспечивает её устойчивость и сохранение всех промежуточных результатов.

3) обеспечение взаимодействия. Сервер служит связующим звеном между пользователями и нейросетевой моделью, обрабатывая запросы от frontend-приложения и передавая ответы обратно. Это делает систему гибкой и масштабируемой, позволяя улучшать её взаимодействие с клиентом и поддерживать стабильность при высоких нагрузках.

FastAPI был выбран для реализации backend-сервера благодаря множеству своих преимуществ. Во-первых, его поддержка асинхронных операций позволяет эффективно обрабатывать большое количество запросов одновременно, минимизируя задержки и повышая общую производительность системы. Это особенно важно для обработки сложных и ресурсоёмких операций, таких как инференс и дообучение нейросетевых моделей. Во-вторых, FastAPI автоматически генерирует документацию для всех API-методов, что значительно упрощает процесс интеграции с другими компонентами системы и внешними сервисами. Это позволяет ускорить разработку, тестирование и дальнейшее обслуживание системы, а также минимизировать вероятность ошибок и недочётов при взаимодействии с API. В результате, выбор FastAPI способствует не только высокой производительности, но и улучшению удобства работы разработчиков, а также повышению надёжности и гибкости всей системы.

Ключевым элементом системы стал механизм мониторинга задач в реальном времени, реализованный через WebSocket-соединения. Эта технология обеспечивает двустороннюю коммуникацию между клиентом и сервером, устраняя задержки, характерные для традиционных методов (например, HTTP-polling). Администраторы получают мгновенные обновления о статусе длительных операций, таких как обучение моделей машинного обучения или обработка данных, через постоянное соединение, что повышает прозрачность управления процессами.

WebSocket минимизирует нагрузку на сервер за счёт отсутствия циклических HTTP-запросов и передачи данных только при изменении состояния задачи. Для обеспечения надёжности реализованы автоматическое восстановление соединения. Решение демонстрирует эффективность в сценариях, требующих непрерывного мониторинга, и служит основой для дальнейшего масштабирования системы.

Система также поддерживает набор API-методов, предназначенных для взаимодействия с пользователями и администраторами, что описано в табл. 1. Эти API позволяют интегрировать внешние системы и обеспечивать гибкость в управлении процессами.

Таблица ниже представляет ключевые функции API, которые обеспечивают эффективное управление данными, моделью и процессами обучения. Для администраторов предусмотрены функции загрузки и удаления датасетов, управления чекпоинтами модели, а также инициирования и мониторинга процесса обучения. Пользователи могут взаимодействовать с моделью через инференс, что позволяет им

получать результаты на основе обученных данных. Такая структура API не только упрощает взаимодействие с системой, но и гарантирует чёткое разграничение ролей между пользователями и администраторами, обеспечивая гибкость в настройке доступа и удобство эксплуатации. Это способствует оптимизации рабочих процессов, повышая производительность и улучшая пользовательский опыт.

Таблица 1

Основные функции API

Функция	Описание
Загрузка датасета	Позволяет администраторам загружать датасеты для обучения модели.
Список доступных датасетов	Возвращает список загруженных датасетов.
Удаление датасета	Удаляет указанный датасет из системы.
Загрузка чекпоинта модели	Загружает чекпоинт модели для её восстановления.
Список доступных чекпоинтов	Отображает все сохранённые чекпоинты.
Запуск процесса обучения	Инициирует процесс дообучения модели с заданными параметрами.
Статус обучения	Возвращает текущий статус обучения модели в реальном времени.
Инференс	Обрабатывает запросы инференса через WebSocket.

Для хранения пользовательских данных, обучающих датасетов и чекпоинтов модели используется СУБД TinyDB, простота интеграции которой с языком программирования Python делает её оптимальным выбором для задач, связанных с локальным хранением данных и минимизацией ресурсозатрат.

Комбинация описанных компонентов обеспечивает высокую масштабируемость и адаптивность системы, а также удобство её использования для решения задач кибербезопасности.

Среди возможных направлений практического использования разработанной системы можно выделить следующие:

– интеграция с центрами оперативного реагирования на инциденты информационной безопасности (SOC), позволяющая значительно сократить время анализа и ускорить принятие решений, что особенно важно в условиях высокой динамики киберугроз;

– система может служить учебным инструментом подготовки специалистов по информационной безопасности в сфере моделирования сценариев кибератак и выработки мер реагирования;

– поддержка малых и средних предприятий (МСП), не обладающих значительными ресурсами для обеспечения собственной киберзащиты, которые могут использовать приложение для получения рекомендаций по реагированию на угрозы;

– оценка уровня безопасности и аудит – генерация детализированных регламентов позволяет выявить пробелы в текущих политиках защиты и оптимизировать их в соответствии с лучшими практиками;

– ускорение процесса реагирования на реальные угрозы благодаря оперативности получения рекомендаций и регламентов для описанных сценариев атак, что минимизирует последствия инцидентов и сокращает простой систем;

– использование командами CERT/CSIRT, которые могут использовать данное приложение для автоматизации подготовки регламентов, что повышает оперативность работы и точность предлагаемых решений.

Заключение

Представленные в статье решения демонстрируют возможности современной нейросетевой технологии в области

противодействия кибератакам. Основной акцент работы сделан на создании системы, способной эффективно адаптироваться к быстро меняющимся угрозам и интегрироваться в существующую инфраструктуру.

Разработанная система на базе модели LLAMA 3.2 представляет собой адаптивное решение для противодействия кибератакам, обладающее высокой функциональностью и гибкостью. Одним из основных результатов работы стало создание эффективного алгоритма дообучения модели, включающего этапы предварительной подготовки данных и применения технологии LoRA. Этот алгоритм обеспечивает адаптацию модели к новым сценариям угроз, что позволяет оперативно реагировать на изменения в динамичной киберсреде, а также минимизацию затрат на переобучение.

Реализованная архитектура системы позволяет обеспечить надёжное взаимодействие между пользователями, модулями системы и процессами обработки данных. Использование современных технологий, включая React для построения интуитивно понятного интерфейса, FastAPI для реализации высокопроизводительной серверной логики и TinyDB для хранения данных, создаёт масштабируемую платформу, способную эффективно справляться с широким спектром задач кибербезопасности.

Система успешно решает такие задачи, как загрузка и обработка обучающих данных, дообучение модели, управление чекпоинтами и выполнение инференса. Разработанный набор API с чётким распределением функций и поддержкой асинхронного взаимодействия упрощает интеграцию системы в существующую инфраструктуру и делает её удобной как для администраторов, так и для конечных пользователей.

Разработанные архитектурные решения и алгоритмы могут быть применены для различных задач, связанных с адаптивным управлением, обработкой больших данных и автоматизацией анализа угроз. Практическая ценность работы заключается в создании инструмента, способного обеспечивать актуальную защиту АИС. Представленная система является шагом в развитии адаптивных технологий для обеспечения безопасности информационной среды, открывая возможности для дальнейших исследований и совершенствования подобных решений.

Список литературы

1. Остапенко Г.А., Васильченко А.П. Нейросетевые задачи и компетенции проектной деятельности по созданию защищённых автоматизированных информационных систем // Информация и безопасность. 2023. Т. 26. Вып. 4. С. 579-586.

2. Васильев А.В. Причины роста количества кибератак: анализ технических и нетехнических факторов / А.В. Васильев // Системный анализ и прикладная информатика. 2023. Вып. 3. С. 48-54.

3. LLaMA 3.2: ИИ на ладони — новые возможности для автономной работы с документами и медиа // URL: https://vc.ru/ai/1525220-llama-32 (дата обращения: 22.10.2024).

4. Модуль нейросетевой регламентации мер противодействия кибератакам / Г.А. Остапенко, А.П. Васильченко, А.А. Остапенко, А.А. Ноздрюхин, Д.С. Покудин, Н.Н. Корвяков // Информация и безопасность. 2024. Т. 27. № 2. С. 239-246.

5. Positive Research 2020 // Positive Technologies. URL: https://www.ptsecurity.com/ru-ru/research/analytics/positive-research-2020/ (дата обращения: 20.09.2024).

6. Жарова О.Ю., Чевычелов А.В. Использование методов машинного обучения для классификации вредоносного ПО / О.Ю. Жарова, А.В. Чевычелов // Электрон. б-ки. Электронный журнал: наука, техника и образование. 2018. Вып. 4/2018(22). URL: https://nto-journal.ru/catalog/informacionnye-texnologii/654/ (дата обращения: 22.10.2024).

7. Понкин Д.И. Концепт предобученных языковых моделей в контексте инженерии знаний / Д.И. Понкин // International Journal of Open Information Technologies. 2020. Vol. 8. No. 9. С. 18-29.

8. Загрузка данных для машинного обучения и глубокого обучения // Microsoft Learn. URL: https://learn.microsoft.com/ru-ru/azure/databricks/machine-learning/load-data/ (дата обращения: 27.09.2024).

9. Чепцов М.Н., Сонина С.Д. Модель оптимизации параметра скорости обучения нейронной сети / М.Н. Чепцов, С.Д. Сонина // Сборник научных трудов Донецкого института железнодорожного транспорта. 2021. № 62. С. 28-32.

Воронежский государственный технический университет
Voronezh State Technical University

Финансовый университет при Правительстве Российской Федерации
Financial University under the Government of the Russian Federation

Поступила в редакцию 20.01.2025

Информация об авторах

Жуков Никита Павлович – студент, Воронежский государственный технический университет, e-mail: znp8b00ff@gmail.com

Белоножкин Владимир Иванович – д-р техн. наук, Воронежский государственный технический университет, e-mail: alexanderostapenkoias@gmail.com

Остапенко Григорий Александрович – д-р техн. наук, профессор, Финансовый университет при Правительстве Российской Федерации, e-mail: ost@fa.ru

Васильченко Алексей Павлович – аспирант, Финансовый университет при Правительстве Российской Федерации, e-mail: rainichek@yandex.ru

Остапенко Александр Алексеевич – аспирант, Воронежский государственный технический университет, e-mail: alexostap123@gmail.com

Чапурин Евгений Юрьевич – ассистент, Воронежский государственный технический университет, e-mail: alexanderostapenkoias@gmail.com

NEURAL NETWORK SYSTEM FOR COUNTERING CYBER ATTACKS BASED ON THE LLAMA 3.2 MODEL

N.P. Zhukov, V.I. Belonozhkin, G.A. Ostapenko, A.P. Vasilchenko, A.A. Ostapenko, E.Yu. Chapurin

The article presents a self-learning neural network system for countering cyber attacks based on the LLAMA 3.2 model. The system combines retraining algorithms, a FastAPI backend server, a React frontend application, and the TinyDB database. An algorithm for retraining the model has been developed to ensure its adaptation to new threats and minimize the cost of retraining. The architecture of the system covers the interaction its components, including data processing, inference, and automated process management via the API. The practical value of the work lies in creating a tool capable of providing up-to-date protection of information systems and adapting to dynamically changing scenarios of cyber threats. Development of architectural solutions and algorithms that can be applied to various adaptive management tasks related to big data processing and threat analysis automation. This approach promotes the development of adaptive technologies and increases the effectiveness of information environment protection.

Submitted 20.01.2025

Information about the authors

Nikita P. Zhukov – student, Voronezh State Technical University, e-mail: mnac@comch.ru

Vladimir I. Belonozhkin – Dr. Sc. (Technical), Voronezh State Technical University, e-mail: alexanderostapenkoias@gmail.com

Grigory A. Ostapenko – Dr. Sc. (Technical), Professor, Financial University under the Government of the Russian Federation, e-mail: ost@fa.ru

Alexey P. Vasilchenko – graduate student, Financial University under the Government of the Russian Federation, e-mail: rainichek@yandex.ru

Alexandr A. Ostapenko – graduate student, Voronezh State Technical University, e-mail: alexostap123@gmail.com

Evgeniy Yu. Chapurin – assistant, Voronezh State Technical University, e-mail: alexanderostapenkoias@gmail.com