Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Владимирский государственный университет
имени Александра Григорьевича и Николая Григорьевича Столетовых»

На правах рукописи

Мазурок Дмитрий Валерьевич

АЛГОРИТМЫ ГЛУБОКОГО МАШИННОГО ОБУЧЕНИЯ В СИСТЕМАХ АНАЛИЗА СЕТЕВОГО ТРАФИКА

Специальность 2.3.1 - Системный анализ, управление и обработка информации, статистика

ДИССЕРТАЦИЯ

на соискание ученой степени
кандидата технических наук

Научный руководитель:
Монахов Михаил Юрьевич,
Доктор технических наук, профессор

Владимир 2025

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность темы. Компьютерная сеть обеспечивает взаимодействие множества устройств и сервисов, генерирующих значительный объем трафика. Необходимость поддержания таких характеристик сети, как доступность и производительность обуславливает потребность в эффективных инструментах анализа сетевого трафика.

Несмотря на различие задач систем анализа сетевого трафика, все они являются неотъемлемой частью процесса принятия решений. Сигналы с таких систем обрабатываются людьми, принимающими окончательное решение. Так, системы мониторинга производительности сети позволяют вовремя принять меры по увеличению вычислительной мощности оборудования, не справляющегося в пиковые нагрузки. Системы обнаружения утечек данных позволяют своевременно предотвращать нарушения законодательства, снижать репутационные риски и минимизировать финансовые потери, связанные с утечкой коммерческой тайны.

По мере увеличения сложности и объема информации, стандартные методы обработки и анализа данных, такие как корреляционный анализ, анализ временных рядов и другие методы, становятся недостаточно эффективными. Возникает потребность в более актуальных и высокотехнологичных подходах. Модели и алгоритмы машинного обучения и искусственного интеллекта, несомненно, являются таким подходом, что подтверждается их широким применением, в том числе, в задачах анализа сетевого трафика. Одним из наиболее перспективных подходов машинного обучения является использование нейронных сетей.

В современном мире нейронные сети с высокой скоростью заменяют классические алгоритмы во многих областях, и анализ сетевого трафика не являются исключением. Применение нейронных сетей позволяет, в большинстве случаев, добиться большей точности, производительности и улучшения других важных метрик, используемых при решении конкретных задач. Однако, повышая данные метрики, при использовании нейронных сетей приходится жертвовать

такими свойствами, как прозрачность, или интерпретируемость решений нейросетевого классификатора, критически важным при принятии решений; робастность, или устойчивость к вариативности входных данных и состязательным воздействиям. Кроме того, при изменении распределения данных, например, ввиду изменения аппаратной конфигурации, требуется своевременно адаптировать модель.

В то же время, возможность отследить и понять причину конкретного решения нейросетевого классификатора позволит не просто повысить степень доверия к алгоритму, сделав его работу более прозрачной, но и позволит снизить уровень ошибок: человек, принимающий решение сможет оценить обоснованность решения нейросети и принять более взвешенное итоговое решение.

Характер потока сетевых данных меняется в зависимости от различных факторов, в том числе предсказуемых, таких как день недели, время. Однако, не все события можно предсказать заранее: износ, сбои оборудования или программного сервиса, изменения в законодательстве, состязательное воздействие и другие факторы так же оказывают влияние на распределение трафика. Обученная на одних данных модель нейросети может значительно снижать качество классификации в таких условиях ввиду низкой робастности. При постоянстве изменения характера данных важно вовремя определить такое изменение и инициировать переобучение модели классификатора.

В настоящее время, в большинстве случаев применения нейросетевых классификаторов в системах анализа сетевого трафика не уделяется должного внимания данным аспектам. Это актуализирует тему настоящего исследования.

Степень разработанности темы. Проблемами повышения эффективности принятия решений занимались такие ученые, как Бородачёв С.М., Романова Н.А., Свирина Л.Н., Михайлов Г.С., Абрахам Вальд, Даниэль Канеман, Амос Тверски. Проблемами интерпретации нейросетевых решений занимались такие ученые, как Маршаков Д.В., Гавриков М.М., Amirata Ghorbani, Abubakar Abid, James Y. Zou, David Alvarez-Melis, T. Jaakkola, Kopf L., Nakajima S., Kloft M., Höhne M.M., Barberan C.J., Balestriero R., Baraniuk R.G. и другие. Применение в сетях методов

интеллектуального анализа данных, в частности, нейронных сетей, исследовали такие ученые, как Комашинский В.И., Смирнов Д.А., Амосов О.С., Амосова С.Г., Иванов Ю.С., Коцыняк М. А., Карпов М. А., Лаута О.С., Дементьев В. Е., Riyazahmed A.J., Resende P.A.A., Drummond A.C., Haripriya A.P., Kulothungan K., Tront, J. G., Marchany, R. и другие. Проблемой робастности и проблемой состязательных воздействий на нейросетевые классификаторы занимались такие ученые, как Чехонина Е.А., Жуков В.Г., Колесниченко М.Д., Лапина М.А., Ржевская Н.В., Котляров Д.В., Дюдюн Г.Д., Jandrik Lana, Shuming Jiao, Z. Song, S. Xiang, Ian Goodfellow, W. Zhang, Quan Z. Sheng, A. Alhazmi, Chenliang Li, Shuyang Gu и другие.

Объект исследования – процесс поддержки принятия решений в системах анализа сетевого трафика.

Предметом исследования алгоритмы глубокого машинного обучения, используемые в системах анализа сетевого трафика.

Цель исследования – решение научной задачи повышения эффективности принятия решений в системах анализа сетевого трафика, работающих на основе алгоритмов глубокого машинного обучения.

В связи с поставленной целью решались следующие **задачи**:

1. Проанализировать существующие модели и алгоритмы глубокого машинного обучения, используемые в задачах анализа сетевого трафика

2. Разработать модели и алгоритмы повышения интерпретируемости нейросетевого классификатора анализатора сетевого трафика

3. Разработать алгоритм оценки функциональной устойчивости нейросети анализатора сетевого трафика в условиях искажений входных данных;

4. Выполнить экспериментальное исследование разработанных моделей и алгоритмов.

Научная новизна проведенных исследований и полученных в ходе работы результатов заключается в следующем:

1. Разработан алгоритм оценки интерпретируемости результатов работы нейросети, отличающийся от существующих использованием положительных

заключений модели и новой метрики точности, что позволяет численно сравнивать различные методы интерпретации решений нейросети анализатора сетевого трафика в задачах бинарной классификации редких событий.

2. Разработан алгоритм оценки устойчивости нейросети анализатора сетевого трафика, отличающийся от существующих использованием аугментации тестовой выборки при помощи вариационного автокодировщика с условием (cVAE) для проведения расчетов с учетом степени переобучения, позволяющий оценить пригодность использования классификатора в условиях зашумленности данных.

3. Предложен модифицированный в части представления и анализа данных алгоритм интерпретации решений классификатора анализатора сетевого трафика, отличающийся от существующих использованием большой языковой модели (LLM), что позволяет корректировать ошибочные решения модели.

На защиту выносится:

1. Расширенный метод интерпретации позволяет снизить ошибку первого рода и повысить общую точность системы обнаружения сетевых вторжений на основе нейросетевого классификатора в среднем на 20% и более.

2. Использование LLM и мультиагентного подхода в совокупности с расширенным методом интерпретации позволяет повысить общую точность классификации нейросети NIDS без привлечения человека на 23% и более.

3. Методика оценки робастности нейросети IDS позволяет оценить стабильность точности работы модели в условиях зашумленности входных данных.

Теоретическая значимость работы заключается в предложенном подходе оценки интерпретируемости нейросетевого классификатора трафика, позволяющем производить численное сравнение разных методов интерпретации.

Предложенный в работе алгоритм оценки устойчивости нейросетевых классификаторов позволяет выбрать наиболее стабильную в реальных условиях эксплуатации модель.

В диссертационной работе предложено совершенствование существующего метода интерпретации, а также расширено применение больших языковых моделей, что в совокупности позволяет корректировать ошибки классификации.

Практическая значимость работы:

Предложен модифицированный в части представления результатов алгоритм интерпретации решений нейросетевого классификатора сетевой IDS, использующий LLM, позволяющий повысить итоговую точность системы более чем на 23%, а также разработанное и зарегистрированное ПО для его применения (свидетельство о государственной регистрации программы для ЭВМ № 2024682393), ПО для проведения эксперимента по оценке повышения интерпретируемости решений нейросетевого классификатора (свидетельство о государственной регистрации программы для ЭВМ № 2024682916), модуль оценки значимости по результатам эксперимента (свидетельство о государственной регистрации программы для ЭВМ № 2024682642). Кроме того, были разработаны модуль оценки актуальности факторов защищенности детектора и классификатора и выдачи адаптивных рекомендаций по повышению защищенности» (свидетельство о государственной регистрации программы для ЭВМ №2021618487); модуль для тестирования скорости работы и точности классификаторов (свидетельство о государственной регистрации программы для ЭВМ №2021618197), модуля аудита нейронных сетей на подверженность состязательным атакам (свидетельство о государственной регистрации программы для ЭВМ №2021619081) и графический интерфейс модуля аудита нейронных сетей на подверженность состязательным атакам (свидетельство о государственной регистрации программы для ЭВМ №2021618155).

Предложенные в работе алгоритмы и средства позволяют повысить качество нейросетевого классификатора от 1,23 до 4,4 раз в сравнении с типовой реализацией классификатора, что подтверждается проведенным экспериментальным исследованием.

Методология и методы исследования. Научные положения работы теоретически обосновываются при помощи системного анализа, аппарата

математической статистики, методов обработки и анализа данных. Для экспериментальной проверки работоспособности предложенных алгоритмов использовалось разработанное программное обеспечение.

Соответствие паспорту специальности. Проблематика, исследованная в диссертации, соответствует пунктам 2, 3, 12 паспорта специальности 2.3.1 «Системный анализ, управление и обработка информации, статистика».

Достоверность и апробация. Степень достоверности результатов исследования подтверждается: рядом экспериментов, проводимых на исследуемых моделях с соблюдением требуемых условий случайности, выполненных на экспериментальных установках; положительным результатом практического использования разработанных средств, а также апробацией в печати и на научных конференциях различного уровня. Полученные результаты были представлены на международной конференции «Improving Security of Neural Networks in the Identification Module of Decision Support Systems», 2020 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT), докладывались и обсуждались на II Международной научно-практической конференции «Наука, технологии и инновации: стратегии развития в современном мире», VIII Международной научно-практической конференции «Наука и глобальные вызовы: перспективы развития», IX Международной научно-практической конференции «Наука и глобальные вызовы: перспективы развития», «Глобальные научные тренды: междисциплинарные исследования». Научно-практическая значимость работы подтверждена рецензируемыми публикациями в журналах и в сборниках научных трудов, докладами на научных конференциях международного и российского уровня, а также государственными свидетельствами РФ о регистрации программ для ЭВМ.

Практическая значимость работы подтверждена внедрением её результатов в ООО «ВОЙС КОММЬЮНИКЕЙШН» г. Москва, ООО «ЮКИТЕХ ЛАБ» г. Москва, ООО «НТЦ «СИСТЕМИНВЕСТ» г. Москва, ООО «АйТиАрт» г. Москва, а также в учебный процесс Владимирского государственного университета.

По результатам диссертационной работы опубликовано 14 научных работ, в том числе в международных базах Scopus и Web of Science – 1, в изданиях, рекомендованных ВАК - 3, получено 7 свидетельств о регистрации программы для ЭВМ.

Личный вклад. Все результаты, изложенные в научно-квалификационной работе, получены автором лично. Постановка цели и задач, обсуждение планов исследований и полученных результатов выполнены совместно с научным руководителем.

Структура и объем диссертационной работы. Диссертация состоит из введения, трех глав, заключения, списка обозначений и сокращений, списка использованных источников из 123 наименований, 9 приложений и содержит 116 страниц основного текста, иллюстрированного 27 рисунками, содержит 4 таблицы.

1 АНАЛИЗ ОБЪЕКТА ИССЛЕДОВАНИЯ

В данной главе приводятся объект и предмет исследования. Анализируются сферы использования нейросетевых анализаторов в компьютерных сетях, изучаются проблемы, возникающие при использовании нейросетей, а также описываются составляющие качества нейросетевых классификаторов, используемых при принятии решений в системах анализа сетевого трафика. Уточняются задачи исследования.

1.1 Объект и предмет исследования

Объектом данного диссертационного исследования является процесс поддержки принятия решений в системах анализа сетевого трафика (NTA, Network Traffic Analysis). Системы анализа сетевого трафика представляют собой инструменты и технологии, используемые для мониторинга, измерения и анализа сетевого трафика в целях управления, безопасности и оптимизации сетевых ресурсов.

Приводят следующие определения анализа сетевого трафика:

1. Метод мониторинга доступности и активности сети для выявления аномалий, включая проблемы безопасности и эксплуатации [1].

2. Практика мониторинга сетевого трафика для получения информации о потенциальных угрозах безопасности и других ИТ-проблемах [1].

3. Совокупность информационно-технологических и инженерно-технических мер по приёму и обработке трафика [2].

Системы анализа сетевого трафика находят применение в различных типах сетей, в частности, в корпоративных и общедоступных сетях. Основные функции таких систем включают обнаружение и устранение неисправностей, обеспечение безопасности сети, оптимизация производительности и планирование масштабируемости.

Системы анализа сетевого трафика являются важным компонентом систем поддержки принятия решений. Эти системы собирают, анализируют и интерпретируют данные о трафике, что позволяет ответственным лицам, таким как сетевые администраторы, руководители ИТ-инфраструктур, специалисты по сетевой безопасности принимать обоснованные и своевременные решения.

Системы анализа сетевого трафика можно разделить на следующие группы:

1. Системы обнаружения вторжений (IDS, Intrusion Detection System) и системы предотвращения вторжений (IPS, Intrusion Prevention System).
2. Системы мониторинга производительности сети (NPM, Network Performance Monitoring).
3. Системы анализа трафика на основе глубокого пакетного анализа (DPI, Deep Packet Inspection).
4. Системы управления сетевым трафиком.

По определению, система обнаружения (предотвращения) вторжений — это программный продукт или устройство, предназначенное для выявления (и блокирования) несанкционированной и вредоносной активности в сети или на отдельном хосте [3, 4]. IDS/IPS являются частью телекоммуникационной сети (наряду с другими средствами защиты информации). События, генерируемые системой, непосредственно используются при принятии решений сетевыми администраторами и специалистами сетевой безопасности.

Большинство известных IPS/IDS с точки зрения идентификаторов обнаружения подразделяются на «Обнаружение вторжений на основе сигнатур (SIDS/SIPS)» и «Обнаружение вторжений на основе аномалий (AIDS/AIPS)». SIDS/SIPS, также известные как «обнаружение вторжений на основе знаний», основаны на идее определения сигнатуры для шаблонов воздействий. Преимуществом таких систем является высокая эффективность обнаружения известных шаблонов сетевых воздействий благодаря сигнатурам. В то же время, этот метод не способен обнаруживать сетевые воздействия нового типа из-за отсутствия их в базе. Так же, в случае использования объемной базы сигнатур, скорость обнаружения может снижаться снижается. AIDS/AIPS, также называемые

«идентификаторами на основе поведения», основаны на идее четкого определения профиля нормальной деятельности. Любое отклонение от этого нормального профиля будет рассматриваться как аномалия, или ненормальное поведение. Основными преимуществами AIDS являются его способность обнаруживать неизвестные и новые типы вторжений и индивидуальный характер нормального профиля активности для различных сетей и приложений. Однако основным недостатком является сложность настройки такой системы, поскольку трудно найти границу между нормальным и ненормальным профилями для обнаружения вторжений [5].

На рис. 1 показана схема использования IPS и IDS систем в сети предприятия.

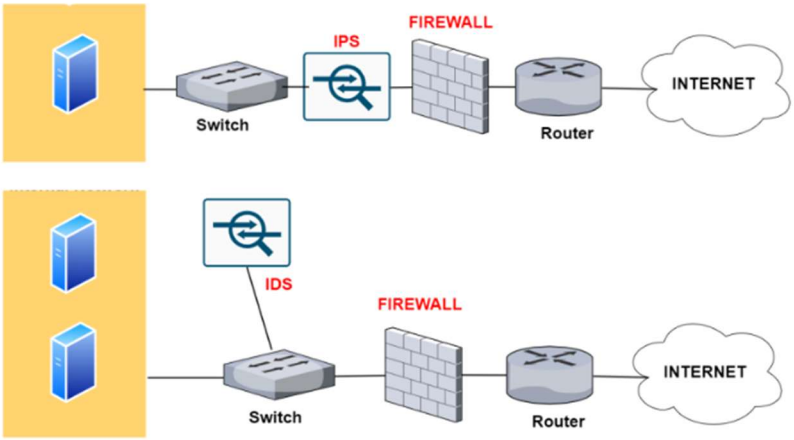


Рисунок 1 - IPS и IDS системы

В настоящее время, в IPS и IDS системам популяризируется использование методов машинного обучения для определения вторжений. Так, в работе [6] приведен сравнительный анализ моделей машинного обучение в задаче обнаружения аномального поведения трафика. В работе Гайфулиной и Котенко [7] проведен сравнительный анализ моделей глубокого обучения для обнаружения аномалий в сетевом трафике интернета вещей, а исследование Антонова и Исаченко рассматривает IDS, основанные на методах машинного обучения, способные к самообучению и эффективному выявлению несанкционированной активности в сетях интернета вещей. Работа Татарниковой и Богданова [8] описывает подход к обнаружению вторжений на различных уровнях архитектуры

сети интернета вещей, основанный на методах машинного обучения, что предлагается использовать при проектировании систем IDS.

В работе [9] исследуются алгоритмы машинного обучения, способные автоматически изучать динамику процессов и стратегии управления в промышленных системах управления (ICS). Авторы отмечают, что такие подходы позволяют создавать детекторы аномалий быстрее и проще по сравнению с методами, основанными на физике и дизайне систем. Однако они также подчеркивают существующие вызовы и проблемы при внедрении этих детекторов в масштабных промышленных сетях.

В исследовании [10] предлагается новая система обнаружения вторжений под названием MAFSIDS, которая использует глубокое Q-обучение (DQL) и метод многоагентного отбора признаков (MAFS) для улучшения выбора признаков и классификации сетевых вторжений. Авторы подчеркивают, что их подход повышает точность обнаружения вторжений в сетях, сохраняя при этом конфиденциальность данных и обеспечивая низкую задержку.

В работе [11] представлена основанная на глубоком обучении. Авторы используют комбинацию сверточных нейронных сетей CNN изучают влияние количества слоев и нейронов на производительность модели. Разработанная модель обнаружения вторжений достигает точности 99,55%.

Работа [12] посвящена применению методов машинного обучения и глубокого обучения для построения IDS в программно-определяемых сетях (SDN). Авторы рассматривают использование этих методов для обнаружения вторжений, таких как DDoS и DoS, и проводят классификацию существующих подходов на обучение с учителем, обучение без учителя, глубокое обучение. Большой вклад работы заключается в детальном сравнении методов, анализе сильных и слабых сторон различных подходов, а также в выделении ключевых вызовов, таких как интеграция ML-алгоритмов в архитектуру SDN и необходимость масштабируемости.

В работе [13] авторы предлагают основанную на глубоком обучении систему обнаружения вторжений для программно-определяемых сетей (SDN). Они

используют автокодировщик совместно с LSTM для обнаружения аномалий в сетевом трафике. Производительность системы оценивается на основе набора данных NSL-KDD, и результаты показывают, что система эффективна при обнаружении вторжений с точностью (precision) 95,11% и recall 96,07%. Набор данных NSL-KDD очень часто используется в исследованиях в области обнаружения сетевых вторжений, и был представлен в 2009 году [14].

В статье [15] предложена модель системы обнаружения вторжений, основанная на архитектуре LSTM, обучаемая на данных из набора KDD Cup 1999 [13]. Результаты тестирования подтверждают, что подход с использованием глубокого обучения эффективно справляется с задачей обнаружения сетевых вторжений, демонстрируя высокую производительность.

В работе [16] предложен подход к распознаванию аномалий для снижения затрат на диагностику цифрового оборудования с использованием двух типов автокодировщиков: глубокого автокодировщик прямого распространения и сверточного. В качестве базы для экспериментов использован набор данных NSL-KDD [13], содержащий сетевые метрики и данные о нормальном и аномальном трафике. Использование автокодировщиков позволило добиться лучшей производительности в сравнении с традиционными методами, такими как логистическая регрессия и метод опорных векторов.

Для решения задач IDS/IPS могут также применяться генеративно-состязательные сети (GAN). Так, в исследовании [17] предложена инновационная система обнаружения вторжений на основе Модель адаптивно обучается распознавать как обычный, так и состязательный сетевой трафик, что позволяет повысить устойчивость IDS к быстро развивающимся угрозам. Для проверки эффективности использовался набор данных KDD Cup 1999 [13], а результаты оценивались с использованием метрик F1, точности (accuracy), полноты и точности (precision). Работа демонстрирует перспективность использования GAN для повышения адаптивности и устойчивости IDS в условиях современных киберугроз. В работе [18] рассматривается проблема дисбаланса данных в наборах IDS, который снижает точность классификации вторжений из-за предпочтения моделей

большинства классов. Для решения этой задачи предложена модель TDCGAN, основанная GAN, решающая проблему несбалансированных данных. Эксперименты проведены на четырех наборах данных (CIC-IDS2017, CSE-CIC-IDS2018, KDD Cup 99, BOT-IOT) с использованием нескольких алгоритмов машинного обучения, результаты демонстрируют эффективность предлагаемой архитектуры TDCGAN.

Популярными примерами IPS/IDS являются Snort, Suricata, Cisco Firepower, Palo Alto Networks IPS, и Check Point IPS. Все они используют, либо могут использовать методы машинного обучения, в частности, нейросети, в своей работе [19, 20].

Системы мониторинга производительности сети – это:

- программные и/или аппаратные компоненты, подключаемые к сети для контроля за производительностью сетевых компонентов, каналов связи и передаваемым трафиком [21];
- это процесс сбора, проверки и анализа сетевых данных для получения информации о особенностях и закономерностях сетевого трафика [22].

Примерами систем мониторинга производительности являются SolarWinds Network Performance Monitor, Nagios, PRTG Network Monitor, Wireshark, и NetFlow Analyzer.

Такие системы используются для:

- анализа производительности сети, выявления узких мест и оптимизации сети;
- выявления первопричины возникающих проблем в сети, локализации неисправных устройств, ошибок в настройках или аномального поведения;
- планирования необходимых мощностей через анализ исторических данных трафика: администраторы могут оценить потребности в ресурсах заранее, определить периоды пиковой нагрузки и принять обоснованное решение о модернизации сети;
- мониторинга производительности критически важных приложений и сервисов, что позволяет выявить проблемы с задержками, потерями пакетов или

снижением производительности и вовремя принять соответствующие меры по оптимизации.

Нейросетевые классификаторы успешно применяются в NPM. Так, в исследовании [23], авторы расширяют и обобщают подход NPM для сценариев с частичной наблюдаемостью, используя нейросети для точного и эффективного анализа в реальном времени, что позволяет простым прямым проходом нейросети анализировать достижимость гибридных систем при достаточных данных для обучения.

Flowmon [24] использует NPM инструменты для анализа производительности сети с микросекундной точностью, предоставляя данные о времени ответа сервера (SRT), времени приема-передачи (RTT) и других показателях, что способствует точному мониторингу и диагностике сетевых задержек и проблем производительности с применением нейронных сетей.

Системы анализа трафика на основе глубокого пакетного анализа (DPI, Deep Packet Inspection) представляют собой продвинутый метод обработки и фильтрации сетевого трафика, который позволяет детально анализировать содержимое пакетов данных, проходящих через сеть. Примерами систем являются Cisco Systems Deep Packet Inspection, Sandvine, Procera Networks, Allot Communications.

В отличие от традиционных методов, категорирующих трафик на основе адресов и портов, DPI исследует пакет целиком, включая заголовки и данные. DPI может определять, какие приложения генерируют трафик, даже если они используют динамические порты или шифрование. Кроме того, анализируются шаблоны поведения для обнаружения аномалий.

DPI применяют для решения следующих задач:

- обнаружение и предотвращение угроз;
- управление и оптимизация трафика;
- соблюдение нормативных требований.

Использование нейросетей в системах глубокого анализа и фильтрации трафика (DPI) становится все более популярным, в особенности, для

классификации и анализа зашифрованного трафика. Так, в работе [25] авторы используют сверточные нейронные сети (CNN) и автокодировщики для классификации сетевого трафика. Эта методика позволяет не только определять типы протоколов, но и выявлять неизвестные или аномальные данные в трафике, что крайне важно для обеспечения безопасности сети.

В другой работе [26] описан подход, представляющий собой рамочную структуру, использующий нейронные сети, в частности, разреженные автокодировщики (SAE) и Глубокие нейросети (DNN) для классификации зашифрованного трафика мобильных устройств.

Системы управления сетевым трафиком представляют собой комплекс технологий и процедур, которые используются для мониторинга, контроля, анализа и управления потоком данных в сетевой инфраструктуре. Примерами являются профайлеры, балансировщики нагрузки (F5 Networks, Citrix NetScaler), системы контроля перегрузок (Riverbed, Cisco Wide Area Application Services), и системы управления полосой пропускания (Cacti, BandwidthD).

Такие системы применяют при решении задач:

- Оптимизации производительности сети (обеспечение максимально эффективной процедуры передачи данных с минимальными задержками и потерями).

- Обеспечение безопасности (защита сети от внешних и внутренних угроз, включая DDoS-атаки, несанкционированный доступ и др).

- Соблюдение политик и стандартов (обеспечение соответствия внутренних политик управления данными и внешних регулятивных требований).

- Управление доступом (контроль доступа к сетевым ресурсам для различных пользователей и устройств).

При этом используются:

- *Политика, основанная на правилах (Policy-Based Management)*: управление, основанное на политиках, определяющих, как трафик должен обрабатываться в зависимости от различных параметров, таких как источник, назначение, приоритет и тип содержимого.

- *Приоритизация трафика (Traffic Shaping)*: регулирование потока данных для определенных видов трафика для гарантии качества обслуживания (QoS, Quality of Service) критически важных приложений.

- *Балансировка нагрузки (Load Balancing)*: распределение трафика между серверами или соединениями для повышения доступности ресурсов.

- *Проактивный мониторинг*: использование аналитических инструментов для наблюдения и анализа трафика в реальном времени с целью выявления потенциальных проблем до того, как они приведут к сбою.

В системах управления сетевым трафиком широко используются нейронные сети. Так, в работе [27] нейронные сети применяются в автоматизации процесса обнаружения злонамерений в реальном времени. Авторы применяют параметро-независимые нейронные сети (WANN, Weights Agnostic Neural Networks) для анализа больших объемов данных, что обеспечивает более высокую точность обнаружение и сокращает количество ложных срабатываний.

В работе [28] описано применение глубоких нейронных сетей (DNN) для точной идентификации аномалий в сетевом трафике. Для этого была разработана двухуровневая система обнаружения аномалий, которая использует DNN в сочетании с анализом ассоциаций для уменьшения числа ложных срабатываний и улучшения точности идентификации вредоносного трафика.

Таким образом, в каждой из представленных категорий систем анализа сетевого трафика применяются нейронные сети, что делает их неотъемлемой частью процесса принятия решений. От результата работы нейросетевых классификаторов напрямую зависит эффективность процесса в целом. Потому алгоритмы глубокого машинного обучения, используемые в системах анализа сетевого трафика, составляют **предмет данного диссертационного исследования**.

1.2 Нейросетевые классификаторы и их особенности

Определим подробнее, что собой представляет нейросетевой классификатор, как вариация нейронной сети.

Нейронная сеть — это модель машинного обучения, вдохновленная структурой и функционированием человеческого мозга. Она состоит из множества взаимосвязанных узлов, называемых искусственными нейронами, которые обрабатывают и передают информацию. Искусственные нейроны получают входные данные, обрабатывают их и выдают выходные данные. Группа одноуровневых нейронов образует слой. Таким образом, в случае полносвязной нейросети, каждый слой получает входные данные с предыдущего слоя (либо исходные данные), обрабатывает их и передает на вход последующему слою [29]. Последний слой формирует прогноз или решение сети (рис. 2), при этом используется, как правило, столько же выходных значений, сколько классов.

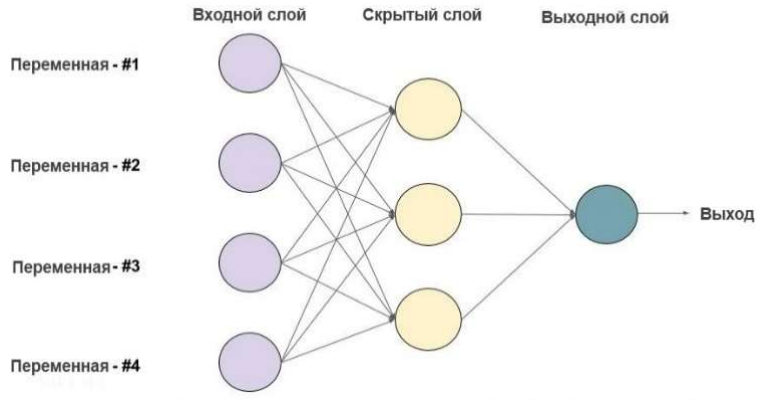


Рисунок 2 - Схема полносвязной нейросети прямого распространения

Обработка, выполняемая каждым нейроном, представляет собой взвешенное суммирование его входных данных, добавление значения нейрона смещения и последующую «активацию» (рис. 3). Активация представляет собой использование нелинейной функция – функции активации. При отсутствии функции активации вся нейронная сеть сводится к одному матричному перемножению не зависимо от ее сложности. Наиболее часто используемые функции активации: Sigmoid [30], ReLU [31] и tanh [32].

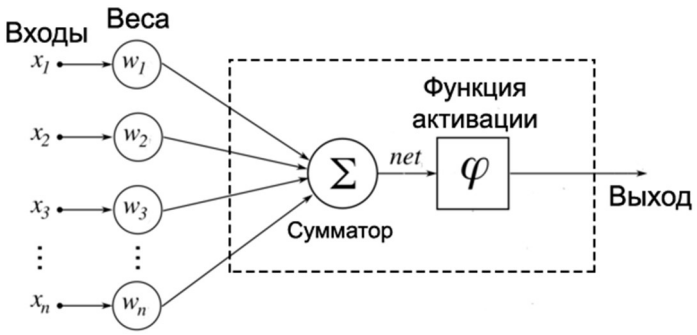


Рисунок 3 - Обработка данных нейроном

Нейронные сети обучаются с использованием больших наборов данных. При этом вначале обучения веса нейронной сети инициализированы, как правило, случайным образом, либо используя метод Xavier [33]. В ходе обучения модель корректирует веса и смещения для минимизации ошибки между ее прогнозами и фактическими результатами. Этот процесс известен как обратное распространение, и он включает в себя обновление весов и смещений в направлении, противоположном градиенту ошибок [32]. Обучение останавливается по достижении желаемой точности, либо других условиях, например, через заданное количество эпох в случае, если нейронная сеть начинает «запоминать» данные (данное явление называется переобучением) или при ином условии [34]. Общая схема процесса обучения нейросети представлена на рис. 4.

Дополнительно при обучении часто используются такие инструменты, как Dropout, BatchNorm и другие для регуляризации обучения, что позволяет снизить влияние переобучения и получить модель с высокой регуляризацией [35].

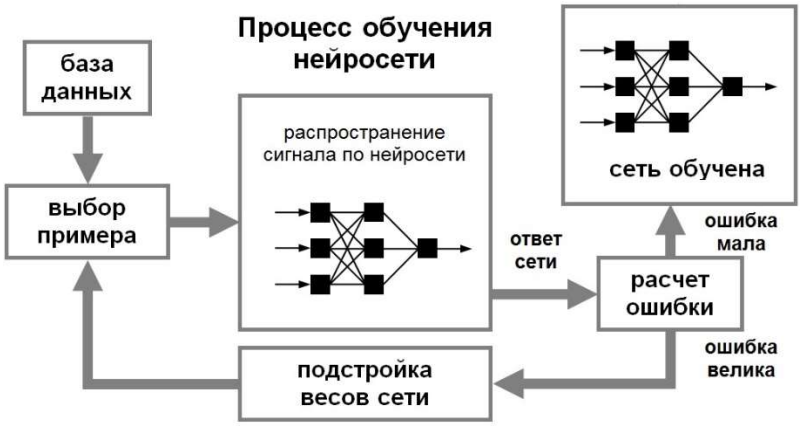


Рисунок 4 - Процесс обучения нейросети

Основными архитектурами нейронных сетей являются:

1. Полносвязные нейронные сети (FFNN – англ. Feed Forward Neural Network, DNN – англ. Deep Neural Network) [29]. Такие сети содержат один и более слоев, все слои соединены в одну последовательность (рис. 2). Такая архитектура часто является составной частью более сложных архитектур нейросетей.

2. Сверточные нейронные сети (CNN) [36]. Такие сети часто применяются к задаче классификации изображений и других данных с пространственной связью между признаками (рис. 5). Параметры в таких сетях представлены в виде фильтров – матриц с весами, образующими в совокупности карты признаков. Данные фильтры применяются по принципу скользящего окна к входному изображению, и возвращают на каждом шаге взвешенную сумму. Полученные данные представляют из себя «каналы» (количество каналов равно числу фильтров) и служат входными данными для следующего слоя. Фильтры, полученные с помощью CNN, формируют определенные признаки - шаблоны из входного изображения, такие как края, текстуры и формы и т. д., повышая с каждым последующим слоем уровень абстракции. За счет применения подхода скользящего окна, такие шаблоны могут быть обнаружены в любой части исходного изображения.

Одним из ключевых преимуществ CNN является их способность автоматически изучать иерархические представления изображения, что делает их особенно подходящими для задач анализа изображений. Кроме того, данный тип сетей обрабатывает информацию параллельно, что значительно ускоряет обучение и работу с уже обученным классификатором.

Рисунок 5 - CNN архитектура

3. Рекуррентные нейронные сети (RNN) [37]. Такой тип сетей часто используется для обработки данных, содержащий временную составляющую, например, классификация текстов, аудио и прочих задач в силу возможности запоминать контекст. В то же время, базовая архитектура RNN редко используются в силу быстрого забывания контекста. Вместо этого предложены улучшенные рекуррентные архитектуры: GRU [38], LSTM [37] (рис. 6). В виду авторегрессивной природы архитектуры, данный тип сетей работает значительно медленнее по сравнению DNN и CNN.

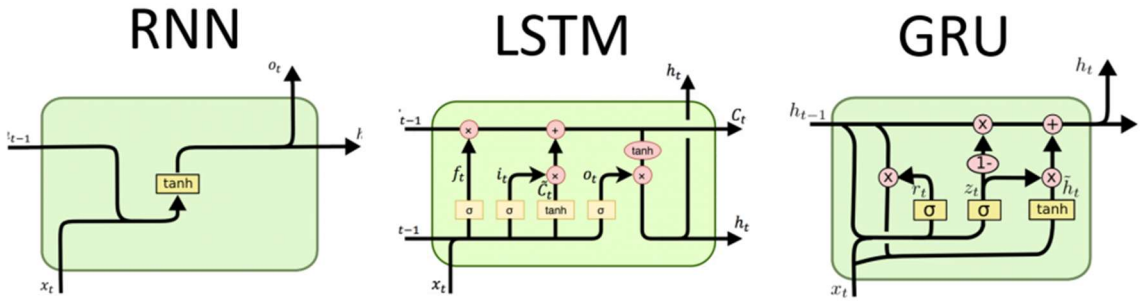


Рисунок 6 - RNN LSTM GRU архитектуры

Получив входные данные, обученная модель нейросети выполняет прямое распространение для получения выходных данных, являющихся итоговым результатом - прогнозом. Прямое распространение включает в себя вычисление активаций для каждого уровня в сети с использованием весов и смещений, полученных на этапе обучения, и входных данных. Окончательная активация выходного уровня — это прогноз, сделанный сетью. Прогноз может быть меткой класса (в случае задачи классификации), непрерывным значением (в случае задачи регрессии) или другим типом данных в зависимости от решаемой проблемы.

В настоящий момент существует множество различных архитектур, привносящих новизну и, как правило, повышающих качество, скорость работы моделей. Одним из примеров может быть архитектура transformer, использующая особый механизм внимания - self-attention [39]. Так или иначе, в основе различных подходов лежат те же описанные выше базовые принципы. Полное описание всех существующих архитектур выходит за рамки настоящего исследования.

1.2.1 Проблема интерпретируемости нейросетей

Интерпретируемость нейронных сетей — сложная и многогранная концепция, которая изучалась многими исследователями в области машинного обучения и искусственного интеллекта. Точного определения интерпретируемости в контексте нейронных сетей не существует, поскольку это может зависеть от различных факторов, таких как тип нейронной сети, данные, на которых она обучается, тип задачи, для которой она используется, а также перспективы и цели вовлеченных заинтересованных сторон [40, 41].

В данной работе будет использоваться определение интерпретируемости, сформулированное в работе [14]: интерпретируемость — способность объяснить или передать значение в понятных человеку терминах. Другое определение представлено в работе [42]: интерпретируемость —это отображение абстрактного понятия в область, в которой человек может разобраться.

Проблема интерпретируемости возникает в связи со сложной структурой модели и стохастическим характером обучения, что делает фактически невозможным точное определение причин того или иного решения нейросетевой модели. Имеющая один и более скрытый слой модель становится «черным ящиком», получающим на вход данные и возвращающем результат. Таким образом возникают проблемы с доверием моделей ввиду непрозрачности алгоритма принятия решений.

Дискуссии об интерпретируемости модели часто сосредоточены на ее математических аспектах. Однако применительно к конкретной задаче обычно требуется дополнительный этап перевода компонентов математической модели в фактические результаты. То есть, результаты, полученные при интерпретации, могут быть понятны только специалисту, в области которой работает нейросетевая модель.

В то же время, не только глубокие нейронные сети имеют свойства «черного ящика». Тип моделей, которые обычно считаются интерпретируемыми математически, могут стать потерять интерпретируемость по мере роста размера

модели. Так, модель линейной регрессии, содержащая малое количество параметров, является интерпретируемой. Однако, едва ли можно назвать интерпретируемой линейную регрессию с тысячами факторов. Параметры модели сохранят такую же интерпретацию, но построение эвристического объяснения на основе предмета исследования в значительной степени затрудняется. То же самое относится к деревьям решений, когда они становятся достаточно глубокими.

Такие подходы машинного обучения как ансамблирование классических интерпретируемых моделей, например, деревьев решений, являются эффективными средствами классификации и регрессии. Однако они менее интерпретируемы по сравнению с одиночными деревьями решений и логистической регрессией. Это обуславливает необходимость использования дополнительных методов интерпретации. Такие методы, как правило, могут быть применены к любым моделям машинного обучения, в том числе и к глубоким нейронным сетям.

Интерпретация глубоких нейронных сетей имеет ряд уникальных проблем, возникающих из-за структуры модели, прежде всего, ввиду наличия скрытых слоев. Наличие скрытых слоев в значительной степени расширяет возможности нейронной сети как алгоритма машинного обучения с одной стороны, с другой стороны - обуславливает потребность в дополнительных методах интерпретируемости ввиду высокой сложности модели и стохастической природе процесса обучения [43-47].

Классификация подходов к интерпретации глубоких нейросетей. Методы интерпретируемости можно классифицировать по различным критериям. Одним из этих параметров является зависимость от исследуемой модели. В рамках этого критерия можно выделить две категории: модельно-независимые и модельно-зависимые методы.

Инструменты интерпретации для конкретных моделей (модельно-зависимые) ограничены классами этих моделей, например, интерпретация весов в модели линейной регрессии. Методы, не зависящие от модели (модельно-зависимые), можно использовать в любой модели машинного обучения и

применять после обучения. Эти независимые методы обычно работают на основе анализа входных и выходных пар признаков. Такие методы не используют внутренние компоненты модели, в т. ч. веса, структурную информацию.

Использование модельно-независимых методов имеет определенные преимущества по сравнению с методами, специфичными для модели, основным из которых является их гибкость. Это позволяет разработчикам машинного обучения использовать различные архитектуры, в то время как свойство интерпретации будет сохранено с применением одного и того же метода интерпретации. Это повышает абстрактность: все, что основано на результатах интерпретации модели машинного обучения, например, графический или пользовательский интерфейс, также становится независимым от исследуемой модели. Учитывая, что для решения задачи обучается, как правило, несколько моделей и архитектур, использование модельно-независимых подходов к интерпретации имеет значительное преимущество.

Другая характеристика, по которой можно разделить методы интерпретации - это область охвата. Для объяснения индивидуальных прогнозов применяют локальные методы, в то время как для объяснения логики работы модели в целом – глобальные.

В зависимости от типов получаемого результата, методы интерпретации разделяются на группы:

— Сводная статистика признаков: методы интерпретации предоставляют сводную статистику для каждого признака. Некоторые методы возвращают одно число для каждой функции, например, важность функции, или более сложный результат, например, силу взаимодействия парных функций, которая состоит из числа для каждой пары функций.

— Визуализация сводки по функциям: значительная часть сводной статистики по функциям также может быть визуализирована. Кроме того, некоторые сводки функций представляют наибольшую пользу именно в виде визуализации. Частичная зависимость признака (Partial Dependence Plot, PDP)

является таким случаем [48]. Графики частичной зависимости представляют собой кривые, которые показывают функцию и средний прогнозируемый результат.

— Внутренние параметры модели: интерпретация внутренне-интерпретируемых моделей попадает в эту категорию. Примерами являются веса в линейных моделях, древовидная структура (пороги, используемые для разделения) деревьев решений. Границы между внутренними элементами модели и сводной статистикой признаков размыты, например, в линейных моделях, потому что веса являются одновременно внутренними элементами модели и сводной статистикой для признаков. Другой метод, который основывается на внутренних параметрах модели, — это визуализация признаков, изученных в сверточных нейронных сетях. Методы интерпретации, выводящие внутренние данные модели, по определению зависят от модели.

— Примеры данных: в эту категорию входят все методы, которые возвращают примеры данных (уже существующие или созданные), для повышения интерпретируемости модели. Одним из популярных методов является контрфактическое объяснение. Чтобы объяснить предсказание экземпляра данных, метод находит противоположную точку данных, изменяя некоторые функции, для которых прогнозируемый результат изменяется соответствующим образом (например, переворот в прогнозируемом классе). Другой пример — идентификация прототипов предсказанных классов. Чтобы быть полезными, методы интерпретации, которые выводят новые точки данных, требуют, чтобы сами точки данных могли быть интерпретированы. Однако, в то время как метод хорошо применим для изображений и текстов, применение для табличных данных с множеством функций часто не приносит желаемого эффекта – повышения интерпретируемости.

— Внутренне интерпретируемая модель: одно из решений интерпретации моделей черного ящика состоит в том, чтобы аппроксимировать их (глобально или локально) интерпретируемой моделью.

Подходы к интерпретации решений моделей машинного обучения.

На данный момент разработано много подходов, позволяющих получить ту или иную интерпретацию для модели. Рассмотрим основные:

1. LIME - (Local Interpretable Model-agnostic Explanations) является модельно-независимым методом, который позволяет локально объяснить предсказания черного ящика [49]. Принцип работы LIME основан на создании простых интерпретируемых моделей (например, линейных регрессий) для каждой отдельной области данных, чтобы объяснить поведение более сложных моделей, для которых требуется интерпретация. Метод строит аппроксимации модели в окрестностях конкретных предсказаний. Основное преимущество LIME заключается в гибкости — его можно применять к любым моделям, от случайных лесов до нейронных сетей, что делает его универсальным инструментом.

2. SHAP (Shapley Additive Explanations) - является модельно-независимым методом, основанным на теории кооперативных игр, в частности на значениях Шепли [50]. Метод SHAP вычисляет вклад каждого признака в итоговое предсказание модели, что делает его удобным для объяснения как глобальных, так и локальных предсказаний. Однако основным недостатком данного метода является высокая вычислительная сложность, особенно для сложных моделей и больших наборов данных, что существенно ограничивает его применение.

3. Anchors — это локальный метод интерпретации, который предоставляет «якоря» — компактные правила, описывающие решения модели на уровне отдельных примеров [51]. Такие правила являют собой точные и простые для понимания объяснения принятых решений. Преимуществом метода является устойчивость к изменениям в данных.

4. Integrated Gradients (Интегрированные градиенты) — это метод интерпретации для моделей, который позволяет оценить вклад каждого признака в предсказание модели [52]. Он основывается на интеграции градиентов модели по отношению к входным данным, является модельно-зависимым и применим только к моделям с дифференцируемой структурой, таким как нейронные сети.

$$IG_i = (x_i - x_i') \int_{\alpha=0}^{1} \frac{\partial F\left(x' + \alpha(x - x')\right)}{\partial x_i} d\alpha \qquad (1)$$

Основное преимущество интегрированных градиентов — высокая точность интерпретации для сложных моделей и достаточно высокая скорость работы.

5. Partial Dependence Plots (PDP) — это глобальный метод интерпретации, который визуализирует зависимость между предсказанием модели и изменением одного или двух признаков [48]. Он оценивает влияние каждого признака на итоговое предсказание, легко поддается интерпретации и применим для моделей с различными типами данных. В то же время метод не стоит применять при выраженной взаимосвязи между признаками.

6. Permutation Feature Importance (Важность признаков по перестановке) - оценивает важность признаков, измеряя снижение точности модели после случайной перестановки значений конкретного признака [53]. Если перемешивание признака приводит к значительному снижению точности, то признак считается важным. Данный модельно-независимый метод может быть неэффективен при наличии сильной корреляции между признаками.

Описанные выше методы можно представить в виде таблицы, позволяющей облегчить выбор оптимального метода (таблица 1).

Таблица 1 - Методы интерпретации

Метод	Доступ к обучающей выборке	Скорость	Точность работы	Тип модели
LIME (Локальные интерпретируемые объяснения, независимые от модели)	Да	Средняя	Средняя	Любая
SHAP (SHapley Additive exPlanations)	Да	Низкая	Высокая	Любая
Anchors (Якоря)	Да	Средняя	Средняя	Любая
Integrated Gradients (Интегрированные градиенты)	Нет	Высокая	Высокая	Нейросети
Partial Dependence Plots (Графики частичной зависимости)	Нет	Средняя	Средняя	Любая
Permutation Feature Importance (Важность признаков по перестановке)	Да	Средняя	Средняя	Любая

Измерение интерпретируемости.

Существующие на данный момент подходы по оценке интерпретируемости разделяются на две группы:

1. Оценка с точки зрения человека

Методы данной группы связаны с такими факторами:

- тип интерпретации - локальный или глобальный;

- полнота данных и их понятность человеку;

- время, необходимое человеку для понимания интерпретации;

- знания и опыт человека, влияющий на его способность разобраться в результатах интерпретируемости.

При этом возможны два сценария:

а) оценка конечными пользователями

Для оценки эффективности интерпретаций в реальных задачах необходимо провести эксперименты с конечным пользователем. Это позволит проверить работу системы в реальном приложении и убедиться в ее успехе по отношению к поставленной цели. Оценка производительности на основе этой цели является убедительным доказательством успешности использования объяснений. Ключевым здесь является то, насколько эффективно объяснения помогают людям в выполнении задач, например, в принятии решений.

б) оценка прочими людьми

Эксперименты с обычными, специальным образом не обученными людьми. Это позволяет уменьшить затраты на проведение оценки. В идеальном сценарии, подход этой оценки будет зависеть только от качества интерпретации, независимо ее типа и точности соответствующего прогноза.

Влияние различных факторов при оценке полезности интерпретируемости исследователями. Так, в работах [54, 55] людям предлагалось решить реальную задачу без или с использованием методов машинного обучения. Т. е. каждый испытуемый имел целью получить наибольшее количество правильных ответов и мог основывать свои решения в том числе на результатах моделей ML. В работах

[56-59] испытуемых просили обращать внимание на результаты работы ML, включая интерпретации, при этом не оценивалось качество ответов испытуемых.

2. Функциональная оценка

Функциональная оценка не требует проведения экспериментов с участием людей. В этом случае для оценки качества объяснений используется формальное определение интерпретируемости в качестве прокси, такое как глубина дерева решений.

Существуют так же подходы к численной оценке качества интерпретации и интерпретируемости. Так, в работах [37, 56] предлагается использовать размер модели как меру интерпретируемости. С ростом сложности модели ее интерпретируемость падает. Однако, данные подходы актуальны прежде всего для классических методов машинного обучения, таких как деревья решений, так как такие методы считаются интерпретируемыми.

В работе [14] предлагается использование корреляции Спирмена как меры качества предложенной интерпретации в случае с атрибуцией признаков. А в работе [60] представлен подход, именуемый sensitivity-n для оценки методов интерпретации, основанных на использовании градиента.

В другом исследовании [61] был предложен метод, названный RemOve для оценки интерпретируемости на основе оценки важности признаков. Такой подход заключается в замере изменения точности модели при поочередном исключении признаков.

Оценка подходов к интерпретации через предоставление примеров рассматривалась в работах [62-64]. Так, в работе [64] исследователи предлагают оценивать качество интерпретации замеряя репрезентативность предоставленных примеров.

Таким образом, не существует единой меры интерпретации, при этом выбор подхода интерпретации и ее оценки обусловлен типом решаемой задачи (и, соответственно, типом используемых данных), архитектурой модели.

1.2.2 Проблема робастности

В соответствии с определением, робастность — это способность системы или алгоритма поддерживать определенный уровень производительности при нормальных условиях, а также при изменении определенных входных данных или условий окружающей среды [65].

Относительно нейронных сетей, понятию робастность можно дать следующие определения:

- инвариантность предсказаний к небольшим, несистематическим изменениям входных данных [66];

- способность модели поддерживать свою производительность на состязательно измененных входных данных [67];

- способность модели выдавать последовательные и точные результаты даже при столкновении с изменениями во входных данных или при возмущениях [68].

Робастность – важное свойство любого алгоритма. Однако, в виду сложных связей внутри модели, а также особенностей обучения, нейронные сети часто имеют низкую робастность. Это означает, что на практике нейросетевые классификаторы могут показывать низкую точность предсказаний на данных, которые определенным образом отличаются от тех, на которых они были обучены.

Низкая робастность нейронных сетей может быть связана с разными факторами. Одной из главных причин низкой робастности является переобучение. Переобучение происходит, когда модель обучается на недостаточно вариативных данных, что приводит к тому, что она начинает «запоминать» образцы входных данных вместо того, чтобы изучать общие закономерности и шаблоны. Как следствие – небольшие изменения в данных влекут за собой существенное изменение предсказаний и падение точности работы классификатора.

Также напрямую влияет на робастность качество подготовки (предобработки) исходных данных и корректный набор параметров при обучении.

В работе [69] приведен следующий подход математического описания робастности:

$$\forall x \in X,\ \forall \delta \in \Delta\colon \max_{\|\delta'\| \le \delta} f\,(x + \delta') = f(x), \tag{2}$$

Где X – пространство входных данных, Δ – множество разрешенных пертурбаций, f – функция принятия решения нейросети, $\|\cdot\|$ - норма. Данная формула проверяет стабильность решений нейросети на пертурбациях радиуса δ.

Подобный способ представлен так же в более ранней работе [70]:

$$\forall x_1, x_2 \in X\colon \|f(x_1) - f(x_2)\| \le K\,\|x_1 - x_2\|, \tag{3}$$

Где K представляет из себя константу Липшица. Данное описание робастности учитывает линейное отношение изменения ответов нейросети с изменением входных данных.

Для оценки робастности нейросети может применяться константа Липшица, рассчитанная как супремум функции отношения магнитуды изменения в выходных данных нейросети к магнитуде отношения разницы входящих данных:

$$L(f) = \sup_{x_1, x_2 \in X, x_1 \ne x_2} \frac{|f(x_1) - f(x_2)|}{|x_1 - x_2|} \tag{4}$$

При этом, большие значения L определяют большую робастность нейросети.

В работе [71] предлагается измерять робастность по формуле (потом перепишу в виде формулы, а не картинки):

$$rob_A(C, P) = \frac{\int_{L_A}^{U_A} H\left(acc\left(C, P_i^A\right) - \Theta\right) di}{U_A - L_A} \tag{5}$$

где $H(x) = 1$ при $x \ge 0$, иначе – 0.

Для повышения робастности модели существуют различные рекомендации, озвученные в исследованиях.

Так, зашумление обучающих данных может помочь нейросети лучше обобщать знания и справляться с шумными и аномальными данными. Исследование, подтверждающее эту теорию, представлено в статье [72], где авторы показывают, что добавление шума к обучающим данным повышает устойчивость модели к состязательным воздействиям и шуму.

Аналогичным примером может служить аугментация данных. Аугментация данных — это процесс создания новых обучающих данных из существующих

путём их модификации, как правило, носящей случайный характер. Это помогает модели стать менее чувствительной к малозначительным деталям во входных данных. Работа [73] содержит обзор различных техник аугментации данных, а также их влияния на робастность моделей.

Дополнительную пользу может принести использование при обучении синтетических данных, приводящих к ошибке классификации (состязательных данных). В работе [69, 74, 75] исследователи показали, что такой подход может значительно увеличить устойчивость модели к состязательным воздействиям.

Наиболее простые, и в то же время – эффективные методы, это использование регуляризации и нормализации. В работе [35] показано, как такая регуляризация, как дропаут помогает уменьшить переобучение и повысить устойчивость нейросетей к вариациям в данных.

Говоря о робастности модели, важно упомянуть, что этот аспект касается способности модели сохранять стабильную производительность при наличии случайных шумов, аномалий и состязательных воздействий в данных. Однако, если изменения в данных носят системный характер и не являются случайными, это может потребовать переобучения модели для адаптации к новым условиям. Такие системные изменения могут существенно повлиять на актуальность модели и её способность корректно функционировать в измененной среде. Обнаружение таких изменений называется мониторингом и является важной составляющей качества системы [76].

Мониторинг данных включает следующие ключевые аспекты [77]:

1. Статистические изменения: осуществляется мониторинг распределений входных данных (признаковое пространство) для определения смещения (сдвига) данных. При этом используются такие статистические характеристиках, как средние значения, стандартные отклонения и квантили [78].

2. Качество данных: регулярная проверка данных на наличие пропущенных значений, аномалий и выбросов.

Мониторинг модели включает [79, 80]:

1. Производительность: проводится анализ основных метрик точности (например, F1-мера);
2. Стабильность предсказаний: мониторинг распределения выходных значений (результатов классификации).

Регулярный мониторинг данных и модели необходим для своевременного выявления ухудшение качества работы модели. При сдвиге данных или деградации модели классификатора ее необходимо переобучить, или адаптировать с учетом актуальных данных. При этом, перед заменой основной модели на более актуальную может проводиться A/B тестирование.

Мониторинг может быть реализован вручную, интегрирована в систему мониторинга ПО. Так же могут быть использованы готовые инструменты, наиболее популярным среди которых является Evidently AI [81].

1.2.3 Проблема состязательных воздействий

Приведем несколько определений состязательного воздействия:

а) Источник [82]: Состязательное воздействие – это воздействие, состоящее в тонком изменении оригинального изображения таким образом, чтобы изменения были практически незаметны человеческим глазом. Измененное изображение называется «состязательным изображением», а при подаче его на вход классификатора вызывает некорректную классификацию, в то время как оригинальное изображение классифицируется корректно.

б) Источник [83]: Накладывание шаблонов, выученных глубинной нейросетью для одного класса изображений, на другие изображения вызывают ошибки классификации. Подобные манипуляции называют состязательным воздействием.

Графически, типовую состязательное воздействие можно проиллюстрировать следующим изображением (рис. 7).

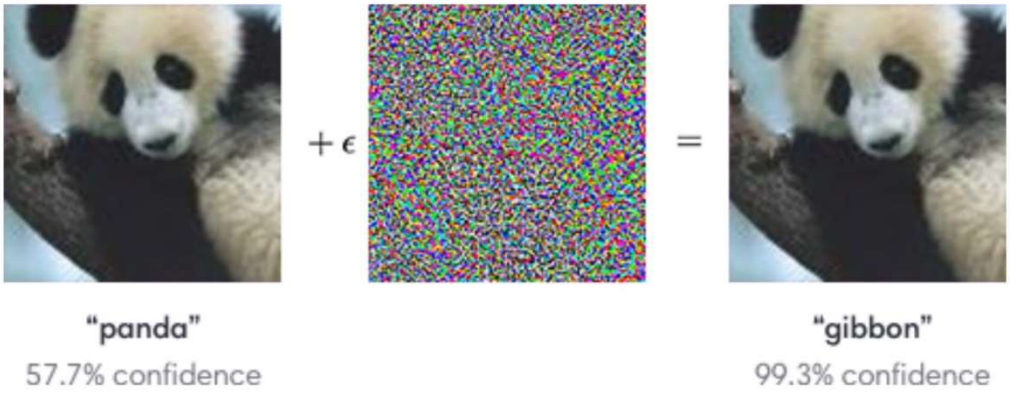


Рисунок 7 - Пертурбация, вызывающая ошибку классификации

На оригинальное изображение (панда), корректно определяемое со степенью уверенности 57,7 %, добавляется точно рассчитанная пертурбация. Будучи незаметной для человеческого глаза, она вызывает ошибку классификации (гиббон) со степенью уверенности классификатора 99,3%.

Рассмотрим основные типы состязательных воздействий.

Тип состязательной фальсификации:

а) Ошибка первого рода – воздействия направлены на возникновение ошибок первого рода классификатором [84, 85]. Например, в задаче поиска вредоносных программ, программное обеспечение, не являющееся вредоносным, классифицируется как вредоносное. В области распознавания изображений, примером может служить состязательное изображение, не поддающееся распознаванию для человека, но классифицируемое «успешно» нейросетью с высокой степенью уверенности.

б) Ошибка второго рода – воздействия, направленные на возникновение ошибок второго рода классификатором. Данный вид воздействий представляет собой данные, «невидимые» для нейронной сети (т. е. не поддающиеся классификации), но потенциально классифицируемые человеком. Одним из примеров может служить внедрение математической закладки в обучающую выборку, что вынуждает нейросеть пропускать данные с особой меткой [83, 86].

2. Знания злоумышленника:

а) Белый ящик – предполагает, что злоумышленник знает все, что относится к тренированной модели, включая тренировочные данные, архитектуру модели,

гиперпараметры, количество слоев, функцию активации, веса модели [87]. Как правило, необходимо иметь доступ к уже натренированной модели, чтобы классифицировать воздействие как воздействие «белого ящика». Большинство состязательный данных генерируется путем расчёта градиентов модели.

б) Черный ящик – предполагает, что злоумышленник не имеет доступа к тренированной модели нейросети, точного знания ее архитектуры, доступа к тренировочным данным. Злоумышленник, как и простой пользователь, знает только выходные данные (результат классификации). Несмотря на то, что большинство состязательных воздействий проводится в условиях белого ящика, они могут быть применены и в условиях черного ящика в связи с принципом: успешное состязательное воздействие на одну нейросеть, с большой вероятностью может быть успешным и с другой подобной нейросетью [88, 89, 90]. Однако, необязательно использовать подобную нейросеть для проведения воздействия черного ящика, т. е. возможным является проведение воздействия без тренировки замещающей нейросети [91].

3.Направленность воздействия:

а) Направленные воздействия представляют собой действия злоумышленника, направленные на создание таких входных данных (изменения входных данных таким образом), чтобы на выходе получить желаемый результат - класс. Например, при классификации лиц, злоумышленник старается создать такие входные данные, чтобы нейросеть определила их как лицо авторизованного пользователя [91].

б) Ненаправленные воздействия имеют своей целью создать такую пертурбацию на оригинальных данных, чтобы нейросеть классифицировала изображение как любой класс, кроме истинного. При классификации лиц задача может состоять в попытке избежать корректную идентификацию личности [91].

4. Частота воздействий:

- единовременное – созданная за одну итерацию пертурбация применяется при проведении воздействия;

- итеративные воздействия – созданная на одном шаге пертурбация используется на следующем для создания более точной пертурбации.

Как правило, итеративные воздействия более успешны, но требуют больше ресурсов для расчетов состязательных данных, времени: имеется необходимость выполнения большего количества запросов к системе воздействия.

Это обуславливает большее распространение единовременных состязательных воздействий по сравнению с итеративными, а также невозможность применения итеративных воздействий при некоторых сценариях.

Пертурбации отличаются по:

1. Универсальность пертурбации – универсальные пертурбации подходят для целого набора данных, в то время как индивидуальные необходимо создавать для каждого изображения отдельно. Создание универсальных пертурбаций является более сложной задачей, так как есть необходимость подбора таких данных, которые одновременно подошли бы ко всем классам сразу.

2. Ограниченность пертурбаций разделяется на:

а) оптимизационная пертурбация представляет пертурбацию как цель проблемы оптимизации: необходимо создать минимально необходимую пертурбацию, чтобы она была незаметна человеческому глазу и, в то же время, вызывала ошибки классификации;

б) ограниченная пертурбация – представляет пертурбацию как ограничение проблемы оптимизации. Данные методы накладывают только ограничения на размер пертурбации;

в) пертурбация без ограничений.

Измерения пертурбаций:

а) $l_p(p \geq 0)$ измеряет магнитуду пертурбации через по формуле:

$$\|x\|_p = \left(\sum \|x_i\|^p\right)^{\frac{1}{p}} \quad (6)$$

l_0, l_2, l_∞ являются соответственно количеством измененных при пертурбации пикселей, Евклидовым расстоянием между оригинальным и измененным

изображением, максимальным изменением всех пикселей в состязательном изображении.

б) PASS - Психометрический перцептивный состязательный показатель подобия (Psychometric perceptual adversarial similarity score) измерят степень подобия оригинального изображения и состязательного в соответствии с человеческим восприятием [92].

в) пертурбации не измеряются.

Причиной возможности проведения состязательных воздействий без доступа к модели и обучающей выборки является, прежде всего, низкая робастность. При наличии доступа к модели основной – алгоритм оптимизации, используемый при обучении, т. е. метод градиентного спуска.

В настоящее время не существует единой формулы для расчета уровня подверженности состязательным воздействиям. Тем не менее, в работе [93] приводится следующая формула для расчета индекса риска состязательного воздействия (англ. ARI - Adversarial Risk Index):

$$ARI(f, \epsilon, X, Y) = \frac{1}{|X|} \sum_{x_i \in X} \max_{||\delta||_\infty \leq \epsilon} \mathrm{l}\left(f(x_i + \delta), y_i\right), \quad (7)$$

где f – функция принятия решения нейросети, ϵ – предел пертурбации, X – тестовые данные, Y- правильные метки класса в тестовых данных, x_i и y_i – тестовый пример и правильная тестовая метка из тестовых данных соответственно, l – функция потерь, δ – небольшая пертурбация, добавленная к данным для создания состязательного примера.

ARI измеряет максимальные потери при всех возмущениях каждого тестового примера в тестовом наборе. Более низкий ARI указывает на более надежную сеть, которая менее подвержена неправильной классификации из-за состязательных возмущений.

Авторы применяют ARI для сравнения надежности различных нейронных сетей, обученных на наборах данных CIFAR-10 [94] и ImageNet [95]. Они обнаружили, что ARI коррелирует с другими показателями надежности может быть

использован для выбора более или менее устойчивых моделей к состязательным воздействиям противника.

Так же можно встретить следующую формулу для расчета риска состязательного воздействия:

$$\mathcal{R}adv = \frac{1}{N}\sum i = 1^{N} \max_{j \neq y_i} \left(\frac{1}{W_j} \sum_{||\delta||_p \leq \epsilon} \mathrm{l}(f(x_i + \delta), j) \right), \quad (8)$$

где $\mathcal{R}_{adv}$ – риск состязательного воздействия, N – число тестовых примеров с правильными метками y_i, W_j – число примеров, которые были классифицированы как j, ϵ – предел пертурбации, l – функция потерь, p – норма для измерения размера пертурбации.

Возможность проведения воздействий, в т. ч. состязательных, на системы анализа сетевого трафика так же изучались исследователями. Рассмотрим примеры с IDS. Так, еще в 1998 году, в работе [96] Ptacek предложили подход к классификации воздействий на IDS и сами воздействия. Классификация включает:

- атака внедрения (insertion attack). Данный тип воздействия учитывает тот факт, что IDS принимает все сетевые пакеты, в отличии от конечной системы.

- атака уклонения (evasion attack). Противоположный атаке внедрения подход. Злоумышленник разбивает пакеты таким образом, чтобы IDS получила только часть из них, опираясь на свойства протокола TCP.

- воздействие отказа в обслуживании (DOS). Классический подход, при котором атакующий выводит на время IDS из строя ввиду исчерпания ресурсов.

Также авторы апробировали воздействия на нескольких коммерческих IDS, при этом все из них оказались уязвимы.

Системы обнаружения сетевых вторжений, использующие нейронные классификаторы, могут оказаться неэффективны в условиях проведения состязательных воздействий, что подтверждается ростом количества исследований на эту тему в настоящее время.

Рассмотрим вначале состязательные воздействия вида «белый ящик». Так, в работе [97] исследователи проверяют возможность проведения состязательных воздействий на IDS сетях IoT. Исследователи провели успешные воздействия, снизив исходную точность классификации с 95,1 до 18 процентов.

В работе [98] авторы применяли состязательное воздействие на различные архитектуры классификаторов, включая CNN и LSTM. При этом, во время воздействия точность моделей снижалась с 98% до 42%.

Так же, в работе [99] авторы успешно провели состязательные воздействия на нейросетевую модель NIDS, экспериментируя с различными подходами генерации состязательных примеров.

Еще больше методов проверено в работе [100].

Интересные результаты представлены в работе Rigaki и Garcia [101], направленной на изучение состязательных воздействий в условиях «черного ящика». Исследователи обучили генеративную состязательную модель нейросети (GAN), модифицирующую исходных трафик таким образом, чтобы быть неотличимым от трафика популярного интернет-мессенджера (для IPS Stratosphere [102], использующей ML для поиска вторжений), и в то же время содержал команды C&C [103]. Таким образом, исследователям удалось успешно передавать инструкции по модификации трафика вредоносного ПО для исполнения команд на зараженном сервере, так, чтобы этот трафик не был заблокирован СЗИ.

В большинстве исследований, проблему восприимчивости к состязательным воздействиям рассматривают как часть проблемы робастности нейросетей. Это обосновывается тем, что модель, восприимчивая к таким воздействиям, в процессе обучения запомнила неробастные паттерны ввиду особенностей данных, процесса обучения. Это делает возможным проведения состязательных воздействий, в частности наложения специальным образом подобранного шума на исходные данные и вызова ошибки классификации у восприимчивой модели. Зачастую, такие изменения насколько незначительны, что достаточно изменить один пиксель на картинке, либо произвести другую незаметную человеческому глазу модификацию, чтобы вызвать уверенную ошибку модели.

В рамках настоящего исследования, проблема состязательных воздействий будет рассматриваться совместно с проблемой робастности с редкими оговорками.

1.2.4 Качество нейросетевого классификатора

В настоящее время отсутствует единое понятие «качественный нейросетевой классификатор», или обособленное понятие качества классификатора. Анализируя данную проблему, можно обнаружить следующие исследования:

В работе [104] качество нейросетевых моделей оценивается через многомерный подход, который учитывает точность модели и её сложность. Точность модели измеряется стандартными методами классификации, в то время как сложность модели рассматривается как мера её интерпретируемости. Авторы предполагают, что более простые модели обычно более интерпретируемы, и используют эту зависимость как критерий для оценки качества. Этот подход позволяет балансировать между точностью и сложностью для оптимизации общего качества модели.

В работе [71] авторы говорят о точности, а также, о робастности модели ANN как о мере ее качества.

В подавляющем большинстве работ под качеством подразумевается мера точности работы модели. Однако, в ходе анализа предмета исследования были отмечены такие проблемы нейронных сетей, как низкая робастность и восприимчивость к состязательным воздействиям, отсутствие интерпретации решений, потребность в адаптации модели. Отталкиваясь от этого, определим критерии качества нейросети и выведем определение, которое будет использоваться в настоящем исследовании.

Под качественным нейросетевым классификатором в работе будет пониматься такой классификатор, который позволяет получать интерпретацию решений, является в высокой степени робастным, в том числе на состязательных данных, и имеет высокую точность и приемлемую скорость классификации, при

работе которого происходит мониторинг за распределением данных, в том числе – его предсказаний с целью своевременной адаптации.

1.3 Формализация задачи исследования

Эффективность принятия решений человеком с использованием систем поддержки принятия решений (СППР) можно оценивать по нескольким ключевым параметрам.

1. *Качество принятых решений* — это самый важный аспект эффективности. Качество можно определить через точность принятия решения, т. е. соответствие принятого решения оптимальному решению существующей задачи с учетом всех релевантных факторов и данных в процессе принятия решения.

2. *Скорость принятия решений*, т. е. время, необходимое ответственному лицу для принятия решения с момента получения информации до момента фактического принятия одного решения. В условиях высоких требований к оперативности (например, в реальном времени для сетевых систем), скорость становится критически важным параметром.

Скорость можно оценить как разницу между временем получения информации из СППР и принятием решения.

Взаимосвязь эффективности принятия решений и СППР представлена на рис. 8. Со стороны СППР, на эффективность принятия решений влияет:

1. *Полнота информации*, т. е. предоставление всей доступной для принятия решения информации.

2. Релевантность выданной информации (если система выдает информацию, не относящуюся к решаемой ситуации, лицу, принимающему решение, потребуется дополнительное время для выбора релевантной информации, а при ее отсутствии – ценность от использования СППР полностью теряется). Релевантная информация – это данные, которые уместны, применимы или имеют решающее значение для конкретной цели, ситуации принятия решений или процесса решения проблем [105].

3. *Достоверность выданной информации* (недостоверная информация может ввести в заблуждение лицо, принимающее решение, что негативно скажется на качестве принятия решения).

4. *Актуальность* и *скорость выдачи информации*, т. е. время, прошедшее с момента возникновения ситуации, в ходе которой возникла потребность в принятии решения, до фактической передачи информации ответственному лицу. В случае невысокой скорости, информация может потерять актуальность.

5. *Доступность для понимания представленной информации*. В случае, если представленная информация сложна в понимании, она может быть неправильно интерпретирована или проигнорирована; кроме того, для работы со сложным представлением требуется дополнительное время при принятии решений.

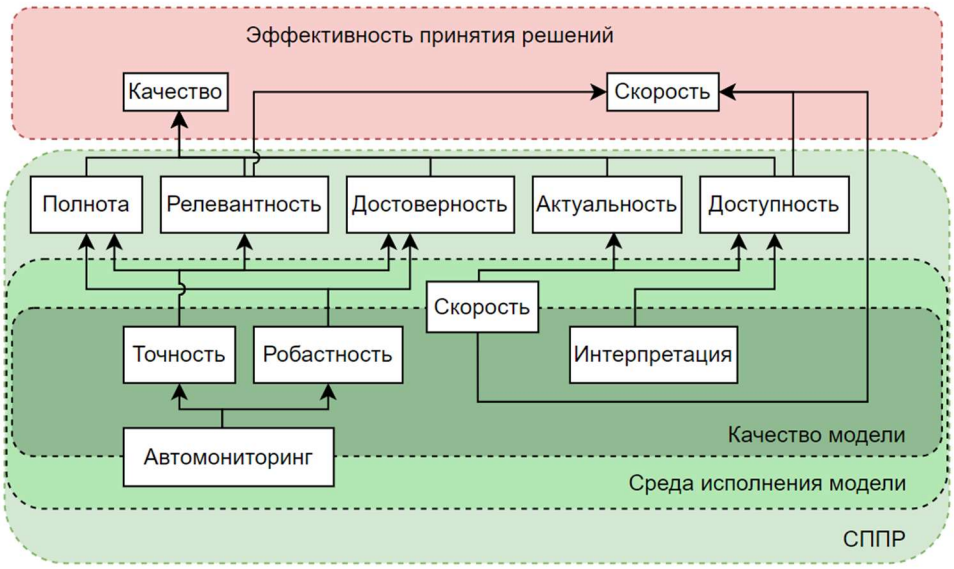


Рисунок 8 - Взаимосвязь эффективности принятия решений и СППР

Эффективность принятия решений в системах анализа сетевого трафикам включает в себя меру качества и скорость принятия решений.

Качество принятия решений можно оценить следующим образом. Пусть имеется множество решений существующей проблемы. При этом имеется СППР, предоставляющая вспомогательную информацию, что позволяет принять наиболее взвешенное и адекватное решение в существующей ситуации. Множество таких решений можно разбить на две группы:

1. S_T – множество решений, являющихся оптимальными в данном случае (при этом для одной проблемы может быть несколько оптимальный, корректных решений).

2. S_F – множество решений, не являющихся оптимальными в данном случае (при этом для одной проблемы может быть несколько неоптимальных, некорректных решений).

В таком случае, качество принятых решений можно оценить через меры точности:

$$Q = \frac{S_T}{S_T + S_F} \tag{9}$$

Значения, близкие к 1 означают высокое качество.

Для оценки точности работы классификатора могут применяться различные формулы, однако в данной работе будет использоваться среднее-гармоническое в виде F1-меры, что позволяет учесть полноту и точность:

$$F = \frac{2 * p * r}{p + r}, \tag{10}$$

где p – точность (precision), r – полнота (recall).

Точность – величина, определяемая по формуле:

$$\begin{cases} p = \dfrac{TP}{TP + FP} \\ p \in (0; 1) \end{cases} \tag{11}$$

Полнота – величина, определяемая по формуле:

$$\begin{cases} r = \dfrac{TP}{TP + FN} \\ r \in (0; 1) \end{cases} \tag{12}$$

Выбор F1-меры обусловлен более оптимальным критерием оценки точности работы нейросетевого классификатора системы анализа сетевого трафика, в которой часто один класс превалирует над другим, т. е. на несбалансированных данных, используемых при тесте классификатора. Кроме того, данная метрика использует и ошибку первого, и ошибку второго рода.

Рассмотрение процесса принятия решений как единого целого может быть дополнительно усложнено введением других мер точности.

В случае, если производится классификация более чем на 2 класса, возможно использование усредненных метрик F1:

- взвешенной:

$$F1_{weighted} = \frac{\sum_{i=1}^{C} w_i F1_i}{\sum_{i=1}^{C} w_i} \quad (13)$$

- макро:

$$F1_{macro} = \frac{1}{C}\sum_{i=1}^{C} F1_i \quad (14)$$

- микро:

$$F1_{micro} = \frac{2 \cdot \sum_{i=1}^{C} TP_i}{2 \cdot \sum_{i=1}^{C} TP_i + \sum_{i=1}^{C} FP_i + \sum_{i=1}^{C} FN_i} \quad (15)$$

Где C – количество классов. *Точностная составляющая оценки защищенности в работе будет именоваться F.*

Рассмотрим пример: система NPM сигнализирует о наличии узла, значительно снижающего доступность ИР в сети. В таком случае, в множестве S_T могут быть такие решения:

1. Заменить проблемный узел на более производительный;
2. Перестроить (оптимизировать) топологию сети.

Множество S_F может быть представлено следующими решениями:

1. Проигнорировать предупреждение СППР;
2. Заменить соседние проблемному узлы на более производительные.

В случае, если ответственное лицо принимает одно из решений множества S_T – его можно считать оптимальным (1), иначе, если решение из множества S_F – оно неоптимально (0).

Таким образом, после N принятых решений будут набраны данные для оценки качества (при этом критически важно иметь возможность оценить оптимальность решений).

Вторую составляющую эффективности принятия решений со стороны человека – скорость принятия решений (D_s) можно оценить как разницу времени от получения сигнала и вспомогательной информации от СППР (T_A) до момента фактического принятия решения (T_D).

$$D_s = T_D - T_A \tag{16}$$

Однако, процесс принятия решения – комплексный и зависит от скорости выдачи информации СППР. Потому важно учитывать общее время принятия решения (D_{total_s}) от момента получения сигнала системой поддержки принятия решений (T_E):

$$D_{total_s} = (T_D - T_A) + (T_A - T_E) \tag{17}$$

$$D_{total_s} = T_D - T_E \tag{18}$$

При этом изменения, вносимые в СППР, влияют на D_{total_s}:

$$\Delta D_{total_s} = \widehat{D_{total_s}} - D_{total_s} \tag{19}$$

где $\widehat{D_{total_s}}$ – время принятия решения после внесения изменений.

Такие изменения могут как ускорять общее время принятия решений (ΔD_{total_s} отрицательна), так и замедлять его (ΔD_{total_s} положительна).

Рассматривая СППР и ее влияние на эффективность принятие решения, следует перечислить такие факторы: полнота информации, релевантность информации, ее достоверность, актуальность, а также доступность и скорость выдачи информации. Измерение данных факторов напрямую значительно затруднено, но может быть оценено через прокси-метрики.

Учитывая, что основу работы СППР составляет нейронная сеть, а точнее – нейросетевая модель классификации, качество работы этой модели (*обозначается в работе* P_c) напрямую влияет на упомянутые выше факторы. Так, чем выше скорость работы модели, тем выше будет скорость выдачи информации в СППР и актуальность, чем выше точность работы – тем выше будет достоверность информации. От дополнительных характеристик СППР, как среды исполнения модели, зависят другие факторы – релевантность, доступность, полнота. Рассмотрим эти факторы.

При эксплуатации системы анализа сетевого трафика постоянно классифицируют данные и получают определенный результат. Как одну из таких систем, возьмем за основу IDS.

Трафик в сети генерируется постоянно. При этом, IDS работает, чаще всего, «окнами»: накопленные за время T_{wind} сетевые данные, которые обрабатываются, из них извлекаются признаки, которые в дальнейшем классифицируются обученной нейросетевой моделью системы. Результат данной классификации – решение, является ли трафик легитимным (0), либо происходит сетевое вторжение (1). Данные результаты используются СППР для помощи оператору в определении, произошел ли фактически инцидент.

В то время как точность модели классификатора IDS высока (*в работе точность обозначается F*), остается шанс ошибки. В то же время, нейронная сеть не предоставляет обоснования своего решения. Принятие гипотезы об истинности решения влечет, потенциально, периодическую частичную или полную приостановку работы сети и, как следствие, снижает доступность. Для более взвешенного решения требуется получить интерпретацию решения классификатора, что позволит сократить ошибку первого рода. Таким образом, при оценке качества классификатора необходимо учитывать возможность получения интерпретации решения. *Оценка интерпретируемости в работе обозначается I*. Также, важно учитывать дополнительное время, затрачиваемое на получение интерпретации конкретного решения модели (*обозначается в работе* T_I).

В то время как сетевой трафик, как правило, хорошо предсказуем, факторы, такие как добавление новых устройств, изменения в законодательстве, другие события могут оказывать значительное влияние на характер данных. Такое изменение, называемое сдвигом данных (при постоянстве изменений), либо шумом (при отсутствии систематики изменений, в т. ч. при изменениях, полученных при проведении состязательных воздействий) может оказать существенное влияние на точность классификации модели нейросети. Такая характеристика как робастность позволит учесть потенциальную подготовленность используемой модели

классификатора к описанным изменениям. При постоянстве таких изменений, т. е. при сдвиге данных, важно своевременно обнаружить сдвиг для последующего инициирования переобучения модели классификатора с целью поддержания высокой точности его работы. Таким образом, при оценке качества классификатора необходимо учитывать также робастность модели нейросети (*оценка робастности в работе обозначается* R), а также использование систем мониторинга сдвига данных (*оценка адаптируемости модели классификатора, учитывающая использование систем мониторинга данных, в т. ч. распределения предсказания модели, обозначается в работе* A).

Необходимо учесть, что при реализации методов повышения робастности может снизиться итоговая точность робастной модели классификатора ($\hat{F}$). Для расчета снижения точности классификации применяется формула:

$$\delta = MAX\left(0, F - \hat{F}\right) \quad (20)$$

В случае, если $\hat{F}$ превышает F, δ примет значение, равное 0, отражающее отсутствие снижения точности.

Последней составляющей качества модели классификатора является скорость его работы на используемом в промышленной среде оборудовании. Низкая скорость работы напрямую влияет на актуальность информации, предоставленной СППР. Скорость работы модели нейросети зависит от реализации кода, среды исполнения, характеристик оборудования, размера модели, разрядности используемых чисел. Если модель анализатора трафика обрабатывает данные ниже скорости реального времени, отставание накапливается, перечеркивая практическую пользу от системы в целом. В работе превышение скорости работы модели реальному времени обозначается C и является, как правило, 1 во всех случаях реальной эксплуатации моделей.

Таким образом, Исходную точность F; Оценку интерпретируемости I; Оценка робастности R; Оценку адаптируемости A; Превышение скорости работы модели реальному времени (C) можно объединить в новое свойство модели –

качество. В данном случае, под моделью подразумевается не только физический файл модели, но и программно-аппаратная среда ее исполнения и мониторинга.

Определим итоговую формулу оценки качества классификатора IDS (по аналогии, данный подход применим к оценке классификаторов других систем анализа сетевого трафика):

$$P_c = C \times \mathrm{F} \times (W_I \times I + W_R \times R + W_A \times A) \quad (21)$$

где W_I, W_R, W_A – веса оценок интерпретации, робастности и адаптируемости соответственно. Общая сумма весов составляет 1; в рамках данного исследования все веса равны $\frac{1}{3}$. T_e определяет оценку повышения времени получения результатов. При этом важно, чтобы время работы модели не превышало реальное время, в то время как общее время работы СППР может его превышать ввиду редкости события – состязательного воздействия.

Максимизация P_c напрямую влияет на качество принятия решений, и, в свою очередь, на общую эффективность принятия решений в системах анализа сетевого трафика, работающих на основе алгоритмов глубокого машинного обучения.

Таким образом, задача исследования заключается в максимизации качества классификатора (P_c). При этом важно учесть следующие ограничения:

Необходимо учесть, что точность исходной модели не может быть ниже определенного порога. Такой порог, помимо прочих факторов, зависит от количества классов, используемых при классификации, точности альтернативных решений. В рамках настоящего исследования будет использоваться порог 0,8 (F =0,8), полученный на основании анализа исследований применимости методов машинного обучения в IDS.

Примем порог в 0,05 для параметра δ, что означает допустимость наихудшей итоговой точности работы классификатора IDS в 0,8 при исходной минимальной точности 0,85.

Обобщим и дополним ограничения:

1. $0 \leq P_c \leq 1$
2. $W_I + W_R + W_A = 1$

3. $0 \leq I \leq 1$
4. $0 \leq R \leq 1$
5. $0 \leq A \leq 1$
6. $0{,}85 \leq F \leq 1$
7. $0 \leq \delta \leq 0{,}05$
8. $\Delta D_{total_s} \leq 30$ секунд

Также необходимо учесть следующее формальное требование:

1. Оператор, принимающий решение, в т. ч. на основе интерпретации, не должен иметь дополнительные знания в машинном обучении (или иных сферах) за исключением краткого инструктажа по работе с интерпретацией.

Для повышения качества классификатора (P_c) предлагается:

1. Реализовать подход расширенной интерпретации решений модели классификатора, позволяющего оператору на основе анализа причин принятия конкретного решения снизить ошибку первого рода, максимизируя тем самым оценку интерпретируемости.

2. Разработать алгоритм оценки функциональной устойчивости нейросети анализатора сетевого трафика в условиях искажений входных данных, позволяющий отобрать наиболее робастную модель.

Выводы к главе 1

1. В результате анализа нейронных сетей определены особенности данного алгоритма машинного обучения, которые необходимо учитывать при использовании: отсутствие интерпретации решений, низкая робастность, восприимчивость к состязательным воздействиям, потребность в адаптации и определении момента необходимости проведения адаптации. Несмотря на широкое применение нейросетей, в настоящее время уделяется недостаточное внимание данным аспектам, что приводит к снижению доверия и функциональной устойчивости алгоритма. В то же время не существует единой оценки

интерпретируемости и робастности нейросетей. Результаты анализа проблем состязательных воздействий, робастности, интерпретации, мониторинга и адаптации нейросетей, определение качества классификатора и его составляющих получены автором лично и были частично опубликованы, в т. ч. в соавторстве (вклад автора более 80%) в [79, 106–109].

2. Выполнена первая задача исследования. Изучены основные сферы применения нейросетей в задачах, связанных с анализом сетевого трафика – это системы обнаружения, предотвращения вторжений; системы мониторинга производительности сети; системы глубокого пакетного анализа трафика, системы управления сетевым трафиком. В работе более детально исследуется применение нейросетевых классификаторов в системах обнаружения сетевых вторжений.

3. Уточнено понятие качества нейросетевого классификатора и выделены составляющие качества нейросетевого классификатора; определено влияние качества модели на эффективность принятия решений. Сформулирована задача оптимизации, заключающаяся в повышении качества нейросетевого классификатора в условиях ограниченности времени получения информации от СППР и отсутствии возможности обучения специалистов, потенциально использующих систему, машинному обучению или иным дополнительным сферам. Повышение качества классификатора можно достичь через реализацию расширенной интерпретации решений модели, а также разработку алгоритм оценки функциональной устойчивости нейросети для выбора наиболее робастной модели. Результаты получены автором лично и опубликованы в [110].

2 ПОВЫШЕНИЕ ИНТЕРПРЕТИРУЕМОСТИ НЕЙРОСЕТЕВОГО КЛАССИФИКАТОРА АНАЛИЗАТОРА СЕТЕВОГО ТРАФИКА

В данной главе описывается ход разработки моделей и алгоритмов повышения интерпретируемости нейросетевого классификатора анализатора сетевого трафика. Описывается порядок и условия проведенных экспериментов и анализ результатов.

2.1 Экспериментальный стенд для исследования повышения интерпретируемости

Для проведения исследования были собраны несколько стендов: для обучения модели, подготовки данных и произведения статистических расчетов, для проведения эксперимента.

Стенд обучения модели

Для обучения модели, подбора оптимальных параметров и выбора оптимальной архитектуры требуются ресурсы, а именно, достаточный объем оперативной памяти, процессор с несколькими ядрами, графический ускоритель.

В рамках исследования был собран стенд на основе операционной системы Ubuntu 18.04, видеокартой NVIDIA RTX 2080TI, центральным процессором Intel Core I7 8700, 32 Гб оперативной памяти DDR3. Для хранения данных использовался SSD Samsung e970 plus evo объемом 1 Тб. Для обучения и тестирования моделей использовался язык программирования Python версии 3.10, фреймворк для глубокого обучения Pytorch 2.0.0+cuda, среда Jupyter. Для обеспечения повторяемости экспериментов были зафиксированы все начальные значения генераторов случайных чисел в библиотеках numpy, torch, random. В torch при обучении использовались только детерминированные алгоритмы, случайные значения в загрузчике данных так же были зафиксированы.

Стенд подготовки данных и произведения статистических расчетов

Для подготовки данных требуется, в основном, производительный центральный процессор и большой объем оперативной памяти. При проведении экспериментов с моделью так же будет полезно наличие графического ускорителя для возможности быстрого проведения экспериментов, требующих использование модели нейросети. Кроме того, наличие быстрого SSD-диска позволит дополнительно ускорить загрузку данных и сократить общее время проведения экспериментов.

В настоящем исследовании был собран стенд с центральным процессором AMD Ryzen 7950X, оперативной памятью 64GB DDR5, графическим ускорителем NVIDIA RTX 3090. Для хранения данных применялся SSD ADATA LEGEND 960 MAX объемом 1 Тб. В качестве операционной системы использовалась Windows 11, в редких случаях – Ubuntu 18.04 и использованием WSL (Windows Subsystem Linux).

Для обеспечения повторяемости экспериментов, так же были зафиксированы все начальные значения генераторов случайных чисел в библиотеках numpy, torch, random. При проведении расчетов и подготовки данных использовался язык программирования Python версии 3.10, фреймворк для глубокого обучения Pytorch 2.0.0+cuda, IDE Jupyter и PyCharm.

Стенд проведения эксперимента

Для проведения эксперимента с привлечением людей было решено использовать выделенный виртуальный сервер провайдера SebekVPS. Сервер использует 4 CPU ядра AMD Ryzen 3900 с производительностью 4,3 Ггц, 8 Гб оперативной памяти DDR4, операционной системой Ubuntu.

Для обеспечения повторяемости результатов экспериментов были зафиксированы все начальные значения генераторов случайных чисел в библиотеках numpy, torch, random. При проведении экспериментов по проверке повышения интерпретируемости применялся язык программирования Python версии 3.10, фреймворк для глубокого обучения Pytorch 2.0.0+cpu, Docker.

При проведении экспериментов также были использованы компьютеры лаборатории кафедры ИЗИ ВлГУ и личные ноутбуки для доступа к API модели с интерпретацией.

Основным фреймворком при обучении и эксплуатации модели выступал Pytorch. В качестве вспомогательных библиотек и фреймворков при обучении использовались:

– Pandas: использовалась для загрузки, очистки и предобработки данных, а также для анализа данных и манипулирования таблицами данных.

– Numpy: применялась для выполнения математических и логических операций на массивах, а также для обработки многомерных массивов данных.

– Matplotlib: использовалась для визуализации данных и результатов моделирования, распределения данных.

– Mlflow: применялась для отслеживания экспериментов, записи и управления результатами.

– Pickle: использовалась для сериализации и десериализации объектов Python, т. е. при сохранении и загрузке моделей и других данных.

– DVC (Data Version Control): применялась для управления версиями данных и моделей, а также для воспроизводимости экспериментов и пайплайнов обработки данных.

– Django для реализации API.

2.2 Разработка алгоритма оценки интерпретируемости

Представим типовой сценарий работы IDS. Модель классифицирует данные на два класса – «воздействие», либо «не воздействие». В таком сценарии самым важным является отсутствие ошибок второго рода, в то время как ложноположительных заключений также не должно быть много.

В случае ошибки второго рода система не справляется со своей задачей, воздействие проходит незамеченным, что недопустимо. При этом ошибка первого

рода вынуждает проводить анализ ситуации, тратя ресурсы предприятия и снижая доверие к системе.

Сигнал от IDS поступает далее к оператору только в случае определения сетевого потока как «воздействие». Таким образом становится понятно, что при оценке качества интерпретации важно рассматривать случаи положительного срабатывания модели – как ложноположительные, так и корректно положительные.

Учитывая отсутствие универсальных метрик интерпретируемости, качество интерпретации можно замерить через прокси-подход. Представим, что оператор, получивший события о вторжении со стороны IDS, имеет возможность дополнительно получить интерпретацию, т. е. объяснение данного решения модели. Тогда, чем более точной, информативной и простой для понимания будет информация, тем больше возможности у оператора скорректировать ложноположительное (и подтвердить корректно-положительное) срабатывание модели.

Таким образом, можно оценить исходную точность модели (F) и скорректированную (оператором) точность ($\hat{F}$). Разница в метриках позволит оценить прирост интерпретируемости. Объединим метрики в оценку интерпретируемости:

$$I = \max\left(0, \frac{\hat{F} - F}{1 - F}\right) \tag{22}$$

Помимо оператора, оценка может быть скорректирована, в частности, большой языковой моделью, промпты для который были специально подобраны для этой задачи. В таком случае можно получить следующую оценку:

$$I_{LLM} = \max\left(0, \frac{\hat{F} - F}{1 - F}\right) \tag{23}$$

Применение LLM позволяет существенно облегчить проведение оценки при достаточной достоверности результатов. Для проведения оценки интерпретируемости результатов нейросетевого классификатора предлагается следующий алгоритм:

Алгоритм оценки интерпретируемости результатов работы нейросетевого анализатора трафика

Шаг 1. Обучить нейросетевую модель NIDS. Записать точность модели F.

Шаг 2. Отобрать данные с ложно и истинно положительными ответами модели.

Шаг 3.1 Сформировать выборку испытуемых с компетенциями, подобными оператору, принимающему решение на основе IDS (не менее 32 человек); объяснить суть исследования и провести инструктаж по работе с интерпретацией и действиями, а именно, выражении согласия или несогласия с ответом модели на основе имеющихся данных для оценки *I*.

Шаг 3.2 Выбрать LLM: GPT4, либо аналогичную, для оценки $\boldsymbol{I_{LLM}}$.

Шаг 4. Используя исправленные ответы, рассчитать скорректированную точность модели $\hat{F}$ (усреднить для нескольких участников).

Шаг 5. Оценить значимость отличия точностей при помощи одновыборочного t-теста с альтернативной гипотезой о повышении точности при использовании метода интерпретации. В случае, если критическое t-значение выше рассчитанного, метод интерпретации принимается неэффективным при выбранном уровне значимости.

Шаг 6. Оценить прирост интерпретируемости за счет использования подхода к интерпретации по формуле:

$$I\ (I_{LLM}) = \max\ (0,\ \frac{\hat{F}\ - F}{1 - F}) \tag{24}$$

Конец.

Для проверки разработанного алгоритма оценки использовался набор данных CSE-CICIDS2018 [111]. Для экспериментов были отобраны наиболее релевантные признаки посредством экспериментов и методов статистического анализа, включены новые признаки. Итоговый набор, содержащий 16232943 строк данных, включает 45 признаков (приложение 1). Обучающий, тестовый и валидационный

наборы содержат 12986354, 1623295, 1623294 строк соответственно. Все обученные модели представляют собой бинарные классификаторы (не воздействие (benign) / воздействие (attack)).

В качестве архитектуры модели выступает полносвязная многослойная модель прямого распространения (DNN). Выбор данной архитектуры обусловлен следующим:

1. При работе с табличными данными (набор данных представляет из себя именно табличные данные), наиболее популярными архитектурами являются: градиентный бустинг, DNN, Архитектура TabTransformer.

2. Применение градиентного бустинга в настоящем исследовании не рассматривалось, т. к. это выходит за рамки исследования.

3. Применение архитектуры TabTransformer предполагает, прежде всего, использование модельно-зависимых методов интерпретации, основанных на значениях внимания (self-attention), что значительно ограничивает применимость данного метода при использовании других архитектур. В то же время, архитектура TabTransformer все равно содержит полносвязный слой для классификации, т. е. можно сказать, что полносвязная сеть прямого распространения является ее частью.

Таким образом, в качестве основной архитектуры при проведении экспериментов была выбрана полносвязная модель прямого распространения с несколькими слоями.

Для подбора оптимальной архитектуры было обучено 30 различных конфигураций моделей, отличающихся числом и шириной слоев, скоростью обучения, регуляризацией и другими параметрами (рис. 9).

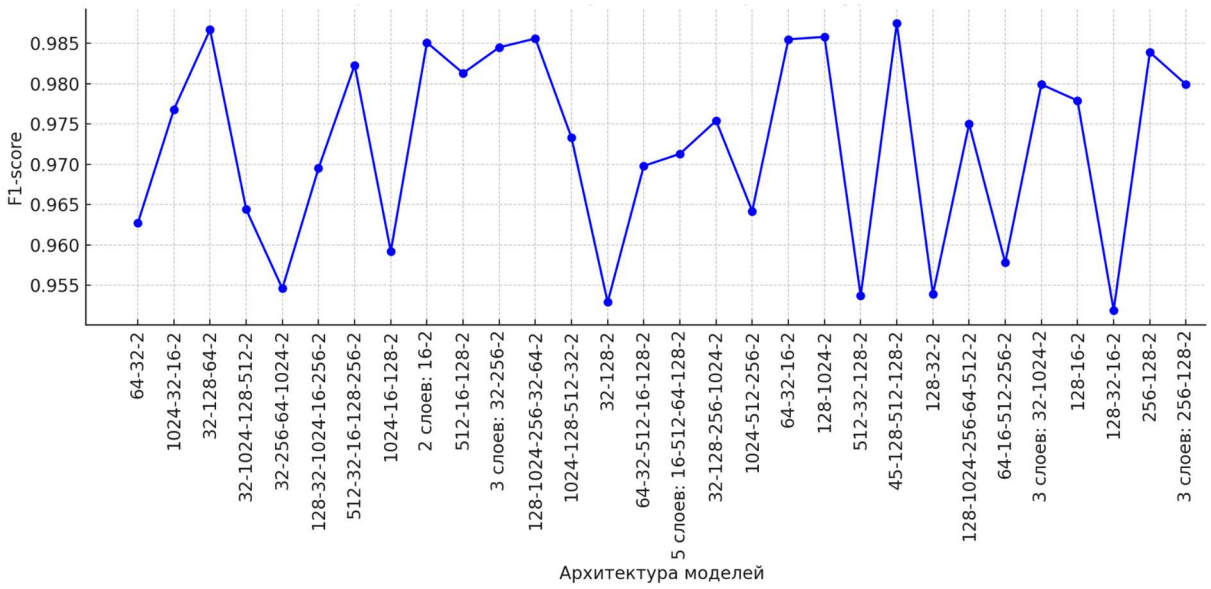


Рисунок 9 - Сравнение F1 различных архитектур

В ходе экспериментов была определена наиболее оптимальная модель с 4 слоями (45-128-512-128-2) и пакетной нормализацией после каждого скрытого слоя. Модель показывает точность на тестовой выборке F1-score 0,9875.

2.3 Разработка расширенного метода интерпретации

Для представления возможности оператору принять обоснованное решение относительно корректности заключения модели, необходимо разработать систему, предоставляющую описание каждого признака, его значение, возможность оценки вклада признака в конкретное предсказание. Учитывая это, в настоящей работе был разработан расширенный метод интерпретации.

В качестве основы был выбран метод IntegratedGradients (IG) ввиду высокой скорости и точности работы.

Для сокращения времени, необходимой для понимания результатов интерпретации, в том числе, для большего понимания произошедшей ситуации, дополнительно к результатам IG были добавлены:

1. Окрашивание в оттенки красного / зеленого цветов в зависимости от значения важности признака;

2. Среднее значение данного признака в подобном случае при отсутствии воздействия с окрашиванием значения в оттенки синего в случае сильного отклонения (более двух стандартных отклонений) текущего значения от среднего;

3. Признаки сгруппированы пол заранее определенным группам, таким как «Флаги в TCP пакетах», «Статистика по времени» и прочее;

4. Каждый признак имеет свое описание, например, признак tot_fwd_pkts имеет описание «общее количество отправленных пакетов»;

5. Имеется поле «Заметки», содержащее дополнительную информацию, зависящую от отличия текущего значения признака от среднего значения, рассчитанного для аналогичного случая (аналогичный набор категориальных признаков, метки класса, одна из 10 групп длительности потока). В случае, если значение отличается более, чем на 2 стандартных отклонения, выводится адаптивное сообщение с возможными причинами данного события.

Такая информация позволит оператору, принимающему решение, в короткие сроки провести анализ сложившейся ситуации и принять адекватное решение.

Кроме того, учитывая высокий уровень развития больших языковых моделей и их потенциальную пользу применительно ко многим задачам, в том числе – задаче анализа интерпретации и представлении дополнительной информации по запросу, в расширенный метод была интегрирована возможность обращения к большой языковой модели GPT4o, отвечающей на вопрос, действительно ли произошло воздействие, а также предоставляющей возможность контекстного взаимодействия.

Для повышения качества ответа большой языковой модели может быть применен мультиагентный подход. В настоящей работе был разработан следующий алгоритм мультиагентного взаимодействия:

Алгоритм взаимодействия агентов LLM для повышения интерпретации

Дано 3 агента: Агент 1 (DLExpert), Агент 2 (NetSecExpert), Агент 3 (Judger).

Шаг 1. Подобрать промпт 1 для Агента 1 таким образом, чтобы Агент 1 решал задачу изучения существующего набора признаков, анализа

интерпретации, определения наиболее характерных комбинаций признаков и важности. Агент должен учитывать возможность модели ошибаться на редковстречаемых комбинациях данных и др.

Шаг 2. Подобрать промпт 2 для Агента 2 таким образом, чтобы Агент 2 решал задачу изучения комбинаций признаков с точки зрения возможных сетевых воздействий, определял характерность подобного набора для каких-либо потенциальных воздействий.

Шаг 3. Подобрать промпт 3 для Агента 3 таким образом, чтобы Агент 3 решал задачу анализа представленных заключений агентов 1 и 2 и генерации окончательного обоснованного заключение. Формат ответа предполагает удобство извлечения итогового решения большой языковой модели автоматическими средствами.

Шаг 4. Передать Агенту 1 промпт 1, данные, значения интерпретаций, ответ модели. Записать ответ.

Шаг 5. Передать Агенту 2 промпт 2, данные, ответ модели. Записать ответ.

Шаг 6. Передать Агенту 3 промпт 3, ответы агентов с Шага 4 и Шага 5.

Шаг 7. Передать ответ Агента 3 оператору; извлечь согласие/несогласие из ответа для расчета метрик.

Конец.

Описанный расширенный метод интерпретации был реализован на языке программирования Python и зарегистрирован (**свидетельство о регистрации ПО №2024682393**).

2.4 Эксперимент по оценке повышения интерпретируемости

Для проведения эксперимента были приглашены 60 студентов ВлГУ кафедры «Информатика и защита информации», обладающие компетенциями, схожими с компетенциями типового сетевого инженера.

Для получения интерпретаций и упрощении взаимодействия с моделью была реализована REST-API, взаимодействующая с описанным выше разработанным

расширенным методом интерпретации. В качестве фреймворка использовался Django, для установки на сервер эксплуатации применялся Docker. Разработанное приложение зарегистрировано (**свидетельство о регистрации ПО № 2024682916**).

Интерфейс представлен на изображении. При открытии адреса приложения требуется ввести идентификатор пользователя, либо войти как администратор (рис. 10).

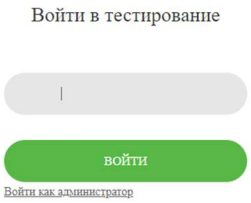


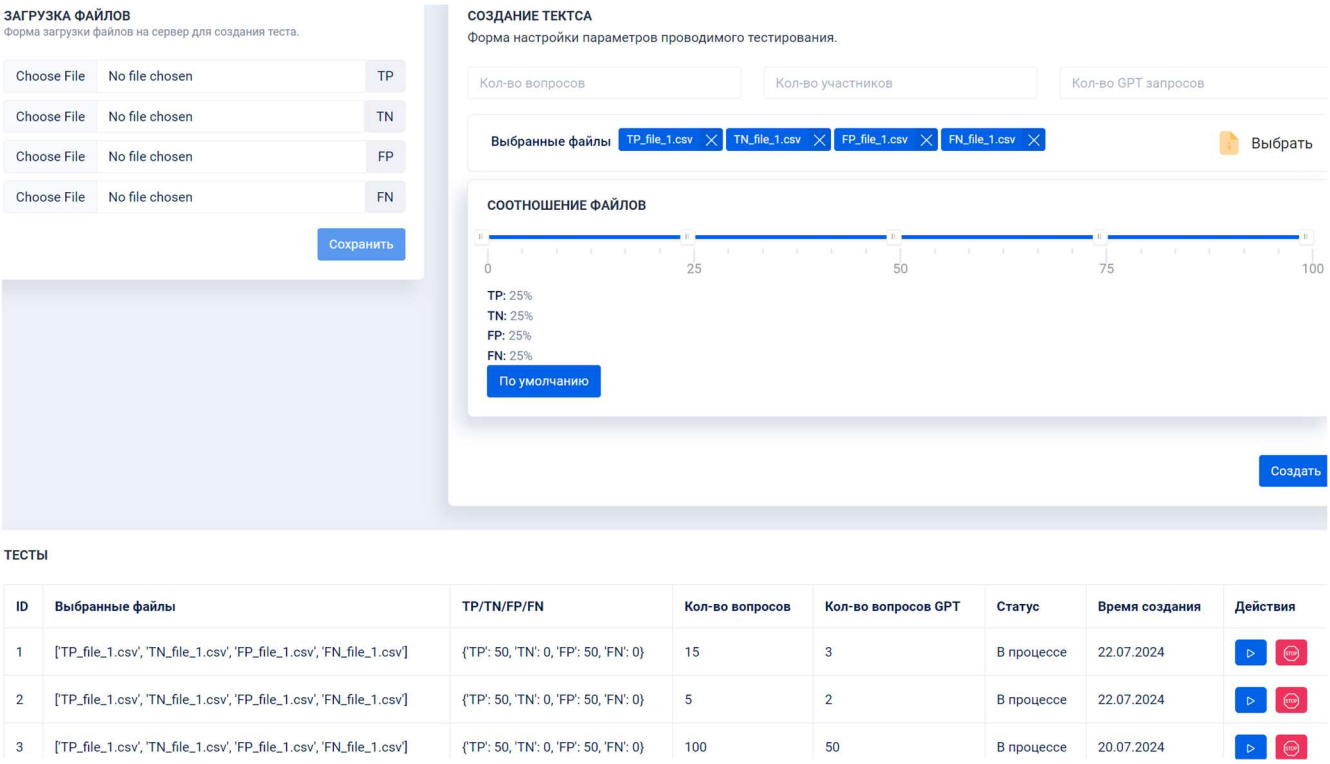
Рисунок 10 - Стартовая страница API

В панели администратора (рис. 11) создается и настраивается эксперимент. В частности, выбираются исходные файлы с данными, разбитые на категории в зависимости от корректности ответов модели:

1. TP - строки данных с действительно положительными заключениями (воздействия);
2. TN – строки данных с действительно отрицательными заключениями (не воздействие);
3. FP – строки данных с ложноположительными заключениями (не воздействие);
4. FN - строки данных с ложноотрицательными заключениями (пропущенные воздействия).

Количество вопросов, их распределение в процентах, количество разрешенных обращений к GPT так же настраивается.

Эксперимент запускается, останавливается централизованно из приложения.



ID	Выбранные файлы	TP/TN/FP/FN	Кол-во вопросов	Кол-во вопросов GPT	Статус	Время создания	Действия
1	['TP_file_1.csv', 'TN_file_1.csv', 'FP_file_1.csv', 'FN_file_1.csv']	{'TP': 50, 'TN': 0, 'FP': 50, 'FN': 0}	15	3	В процессе	22.07.2024	
2	['TP_file_1.csv', 'TN_file_1.csv', 'FP_file_1.csv', 'FN_file_1.csv']	{'TP': 50, 'TN': 0, 'FP': 50, 'FN': 0}	5	2	В процессе	22.07.2024	
3	['TP_file_1.csv', 'TN_file_1.csv', 'FP_file_1.csv', 'FN_file_1.csv']	{'TP': 50, 'TN': 0, 'FP': 50, 'FN': 0}	100	50	В процессе	20.07.2024	

Рисунок 11 - Панель администратора эксперимента API

Каждому пользователю выдается уникальный ID, по которому осуществляется вход в эксперимент, записываются его ответы в базу данных PostgreSQL, из которой после собирается отчет. Для получения идентификатора пользователя, просмотра статистики по эксперименту предусмотрена ссылка на таблицу (рис. 12).

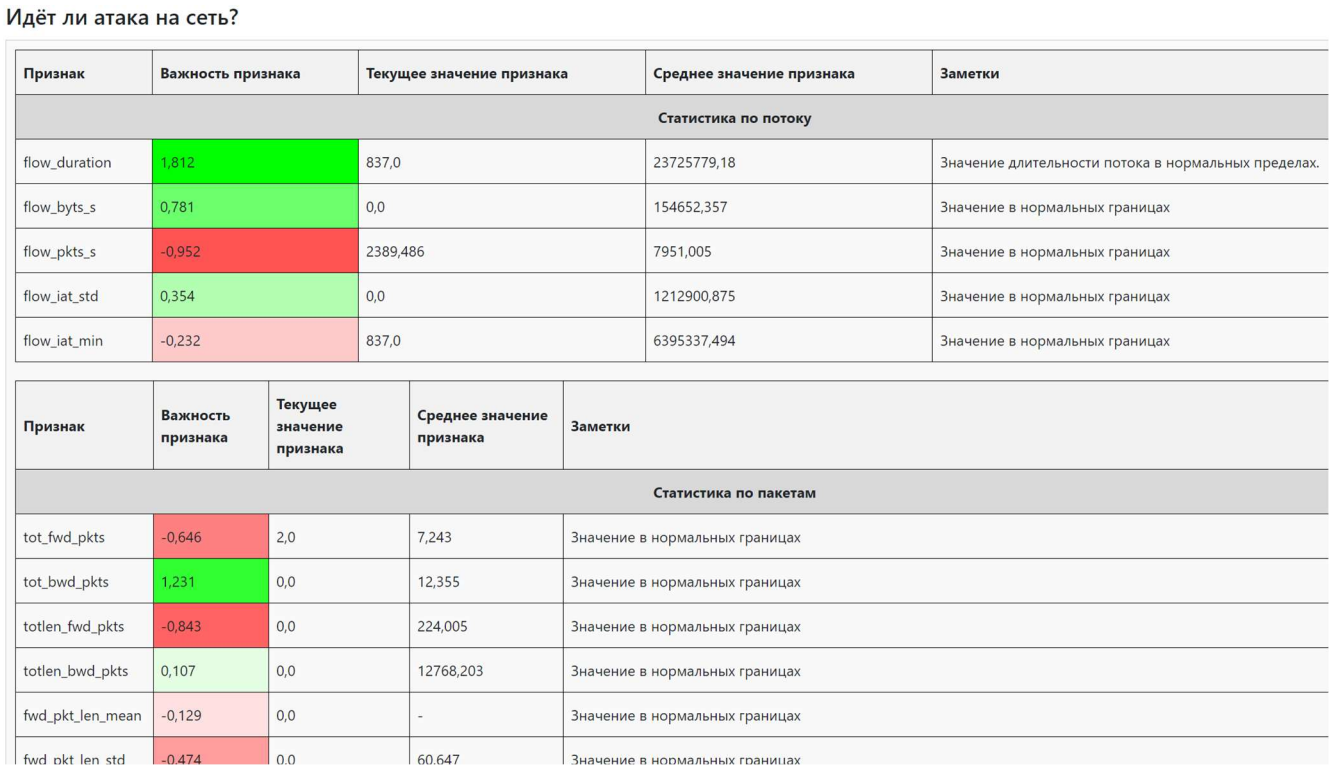
Участники теста № 1

UUID	Кол-во запросов к GPT	Кол-во пройденных вопросов	Статус
vsVz			Онлайн
zqdZ			Онлайн
PcOe			Не активировано

Рисунок 12 - Таблица с идентификаторами пользователя и статистикой

Интерфейс эксперимента представлен на рис. 13. Испытуемому предлагается ответить на вопрос, идет ли воздействие, опираясь на представленные данные. Данные на странице представляют собой расширенный метод интерпретации, отображающий объяснение по конкретной классификации, кнопку открытия чата

с LLM GPT4o. Страница так же включает варианты ответа: «да», «нет», «не знаю», после нажатия на один из которых происходит переход к следующему вопросу.



Идёт ли атака на сеть?

Признак	Важность признака	Текущее значение признака	Среднее значение признака	Заметки
Статистика по потоку				
flow_duration	1,812	837,0	23725779,18	Значение длительности потока в нормальных пределах.
flow_byts_s	0,781	0,0	154652,357	Значение в нормальных границах
flow_pkts_s	-0,952	2389,486	7951,005	Значение в нормальных границах
flow_iat_std	0,354	0,0	1212900,875	Значение в нормальных границах
flow_iat_min	-0,232	837,0	6395337,494	Значение в нормальных границах

Признак	Важность признака	Текущее значение признака	Среднее значение признака	Заметки
Статистика по пакетам				
tot_fwd_pkts	-0,646	2,0	7,243	Значение в нормальных границах
tot_bwd_pkts	1,231	0,0	12,355	Значение в нормальных границах
totlen_fwd_pkts	-0,843	0,0	224,005	Значение в нормальных границах
totlen_bwd_pkts	0,107	0,0	12768,203	Значение в нормальных границах
fwd_pkt_len_mean	-0,129	0,0	-	Значение в нормальных границах
fwd_pkt_len_std	-0,474	0,0	60,647	Значение в нормальных границах

Рисунок 13 - Интерфейс эксперимента

Подход к интерпретации полностью повторяет модифицированный и описанный в главе 2 настоящего исследования подход. Для проверки гипотезы о способности применения большой языковой модели для повышения качества интерпретации, модель GPT4o опрашивается, получая задачу проанализировать исходный набор значений, а также результаты интерпретации. Заключение модели, а также возможность уточняющих вопросов открывается по кнопке «спросить, что думает GPT» (рис. 14).

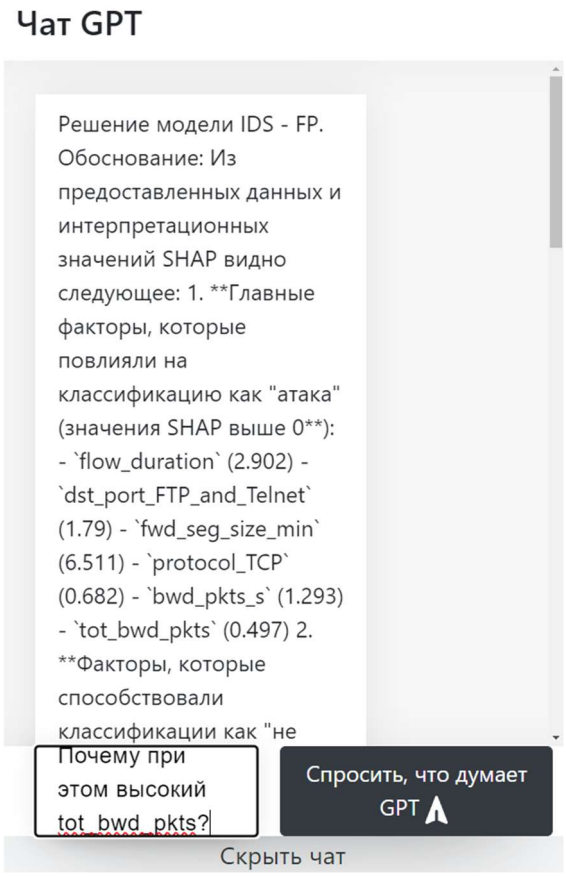


Рисунок 14 - Интерфейс взаимодействия с LLM GPT4o

Для проверки гипотезы о возможности исключения из цепочки принятия решения человека, разработанное приложение реализует опцию проведения эксперимента без участия человека (только на ответах большой языковой модели).

Для анализа результатов было разработано отдельное приложение (**свидетельство о регистрации ПО №2024682642)** (рис. 15).

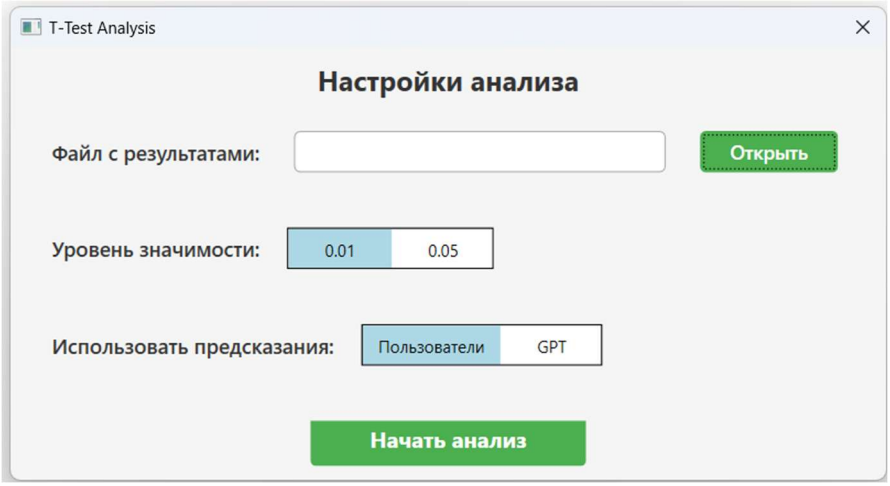


Рисунок 15 - Параметры анализа результатов эксперимента

Для получения результатов возможно как прямое взаимодействие (по вызовам API), так и через загрузку результирующего файла.

Приложение проводит одновыборочный t-тест с заданными параметрами и данными. Пример результатов представлен на изображении (рис. 16).

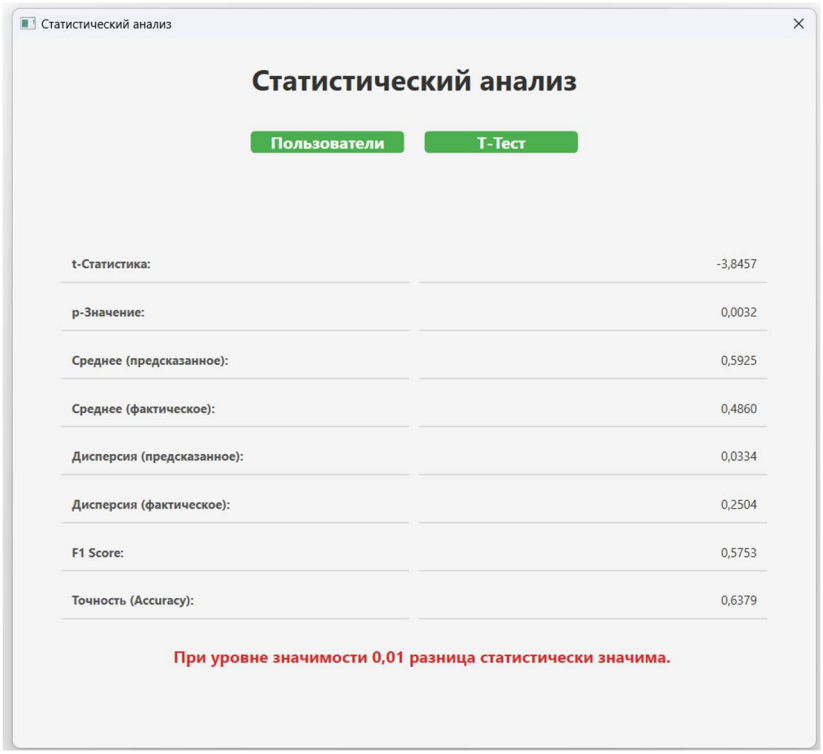


Рисунок 16 - Пример результатов эксперимента

Помимо просмотра общих результатов, разработанное приложение позволят получить статистику точности в разбивке по пользователям (испытуемым) (рис. 17).

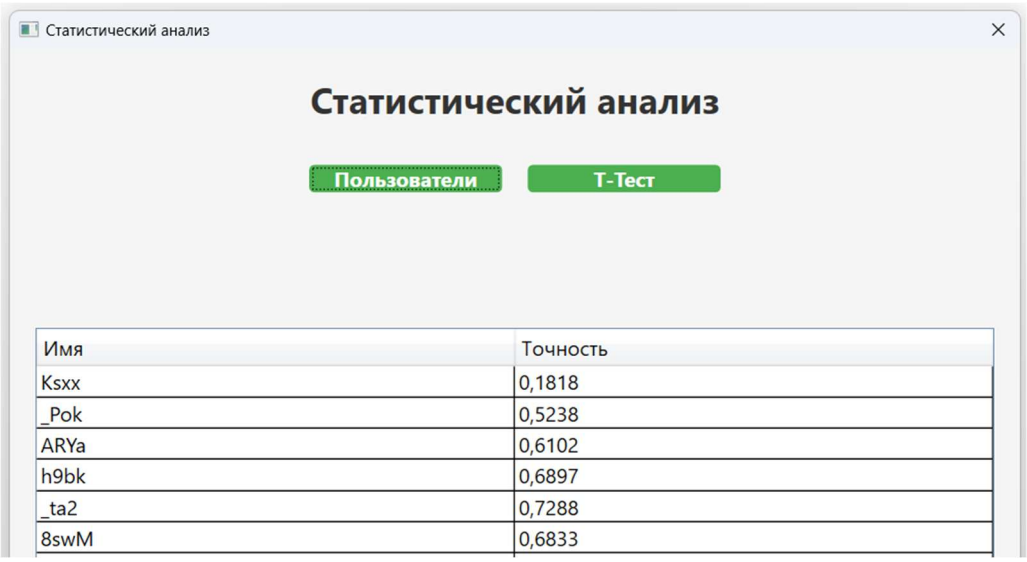


Рисунок 17 - Статистика точности в разбивке по испытуемым

При проведении эксперимента были случайно отобраны 60 тестовых примеров, 30 из которых модель ложно отнесла к классу «воздействие» (FP), 30 – корректно отнесены к классу «воздействие» (TP). Такая постановка соответствует реальной ситуации, когда проверяются только срабатывания модели анализатора сетевого трафика, наиболее частый класс – отсутствие воздействия – не проверяется. Каждому участнику эксперимента предлагалось проверить и, при необходимости, скорректировать ответы модели на всех вопросах. Вопросы передаются в случайном порядке для отсутствия возможности «подсмотреть» ответ у других участников эксперимента, общая же выборка вопросов у всех участников была одинакова.

При эксперименте велась запись времени (округление до секунд) и вручную скорректированных ответов. При этом информация о корректности конкретного предсказания была скрыта. Уровень значимости был выбран равным 0,05 при проведении эксперимента. Так же учитывается ограничение: $\Delta D_{total_s} \leq 30$ секунд.

Эксперимент проходил в несколько этапов:

1. Замер исходной точности модели на отобранных данных. Предполагается, что в отсутствии интерпретации и других вспомогательных инструментов (модель представляет лишь ответ, либо ответ и уверенность в нем) оператор не будет вносить коррективы в работу модели, будет ей полностью доверять. В таком случае $I = 0$, ввиду невозможности принять взвешенное решение на основании лишь ответа модели, и рациональным подходом является полное согласие с ответами модели, т. е. $F = \hat{F}$. В ходе эксперимента начальные условия сравнивались с разработанным расширенным методом интерпретации.

2. Испытуемым предлагается оценить корректность решения нейросетевого классификатора IDS основываясь на:

– Результатах расширенной интерпретации на основе разработанного в рамках данного исследования ПО;

– Результатах заключения большой языковой модели, с возможностью дальнейшего взаимодействия (уточняющие вопросы). При этом не использовался мультиагентный подход.

В рамках данного этапа проверяется гипотеза о том, что расширенный метод интерпретации позволяет снизить ошибку первого рода и повысить общую точность системы обнаружения сетевых вторжений на основе нейросетевого классификатора.

3. Использование только подхода с большой языковой моделью без участия человека. В данном случае проверяется гипотеза, что использование LLM и мультиагентного подхода позволяет повысить общую точность классификации нейросети NIDS без привлечения человека. Использованный алгоритм агентного взаимодействия представлен выше. Подобранные для эксперимента промпты представлены в приложении 2.

Важно подчеркнуть сложность фиксации результатов работы LLM, несмотря на возможность управления температурой, что обуславливает потребность проведения многократных экспериментов. Таким образом, при проведении этапа 3 было отобрано 32 независимых выборки данных, на каждый вопрос из которых отвечал мультиагентный тандем из LLM GPT4o.

Результаты эксперимента

Этап 1

Результаты этапа 1 эксперимента равны точности модели на используемой в эксперименте выборке, а именно, F1-score 0,667.

Этап 2

Результаты этапа 2 эксперимента частично представлены в таблице 2 (с 1 по 21 участника из 60, полная таблица приведена в приложении 3). Испытуемые приступали к тестированию одновременно, эксперимент запускался централизовано через интерфейс администратора.

Перед началом эксперимента был проведен краткий инструктаж. Кнопка «не знаю» была отключена на время проведения эксперимента.

Таблица 2 - Часть результатов этапа 2 эксперимента

Участник	TP	TN	FP	FN	P	R	F1
1	26	21	9	4	0,743	0,867	0,800
2	26	23	7	4	0,788	0,867	0,825
3	26	23	7	4	0,788	0,867	0,825
4	25	24	6	5	0,806	0,833	0,820
5	26	23	7	4	0,788	0,867	0,825
6	26	23	7	4	0,788	0,867	0,825
7	26	23	7	4	0,788	0,867	0,825
8	26	23	7	4	0,788	0,867	0,825
9	25	22	8	5	0,758	0,833	0,794
10	26	22	8	4	0,765	0,867	0,812
11	24	23	7	6	0,774	0,800	0,787
12	26	23	7	4	0,788	0,867	0,825
13	26	23	7	4	0,788	0,867	0,825
14	24	23	7	6	0,774	0,800	0,787
15	23	22	8	7	0,742	0,767	0,754
16	26	23	7	4	0,788	0,867	0,825
17	26	23	7	4	0,788	0,867	0,825
18	25	23	7	5	0,781	0,833	0,806
19	26	23	7	4	0,788	0,867	0,825
20	25	24	6	5	0,806	0,833	0,820
21	27	21	9	3	0,750	0,900	0,818
Ст. откл	1,504	0,976	0,976	1,504	0,028	0,050	0,034
Среднее	24,900	22,783	7,217	5,100	0,775	0,830	0,801
Медиана	25,000	23,000	7,000	5,000	0,788	0,833	0,813
LLM	25	23	7	5	0,781	0,833	0,806

Таким образом, среднее значение F1-score среди испытуемых составило 0,801 (повышение на 20,09% по сравнению с базовым случаем), медианное 0,813. При этом точность ответов LLM составила 0,806 (рис. 18).

Рисунок 18 - F1-Score участников и LLM

Для проверки значимости отличий средних значений точности был проведен одновыборочный t-тест с уровнем значимости 0,05. Было получено значение t-статистики 30,358, подтверждающее статистически значимое отличие средней точности при использовании предложенного подхода интерпретации, что свидетельствует о повышении интерпретируемости ответов нейросети.

Рассчитаем по формулам (21) и (22) оценку интерпретируемости. Полученные в ходе эксперимента оценки интерпретируемости: I=0,403 при использовании среднего значения точности в эксперименте с людьми и I_{LLM}=0,419 при использовании LLM. Таким образом, в ходе эксперимента применение разработанного подхода интерпретации значительно повысило интерпретируемость с 0 до 0,419.

Проверим, как часто участники повторяли ответы LLM, для этого проверим корреляцию Мэтьюса между ответами (рис. 19).

Исходя из анализа корреляции, большинство участников практически полностью повторили ответы большой языковой модели. Средняя корреляция Мэтьюса среди испытуемых составила 0,898, что является статистически значимым с уровнем значимости 0,05 (t-значение составило 118,221, p-значение 8,2e-72).

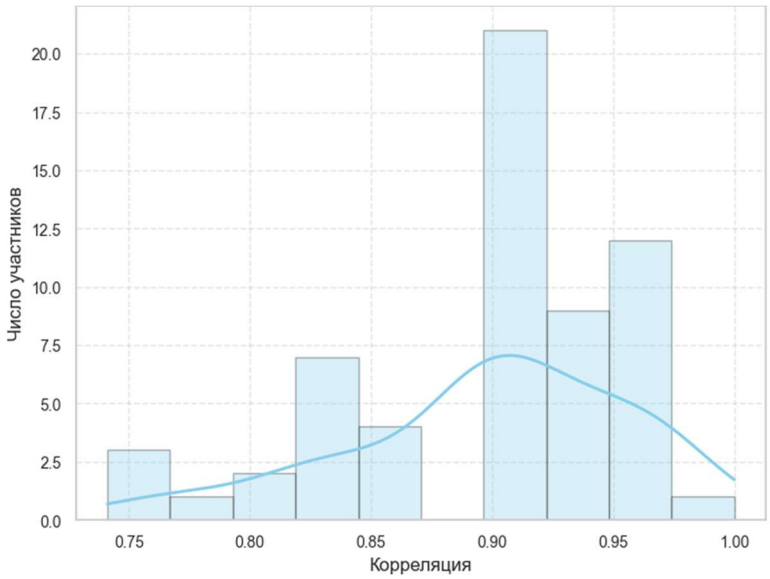


Рисунок 19 - Корреляция ответом участников и LLM

Для оценки соблюдения условия по времени, требуемого для ответа по результатам интерпретации, API записывает по каждому затрачиваемое время. Так, среднее время ответа участников в ходе эксперимента составило 24,31 секунд, что удовлетворяет начальным требованиям (рис. 20). При этом был замечен небольшой тренд на снижение времени на ответ по мере продвижения по вопросам.

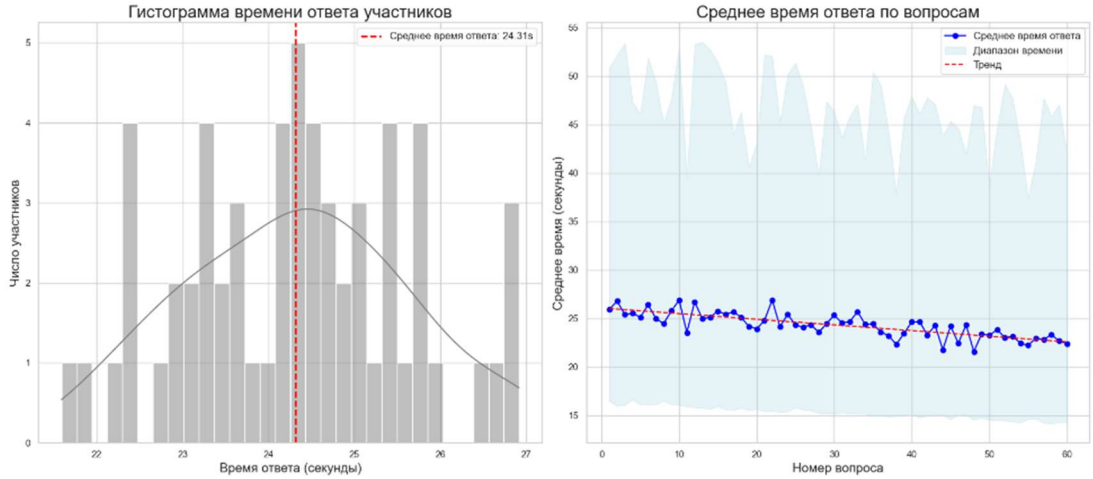


Рисунок 20 - Распределение времени на ответ

Этап 3

Результаты этапа 3 эксперимента представлены в таблице 3. В ходе эксперимента получена средняя точность при использовании мультиагентного подхода F1 = 0,826; средняя скорость ответа на вопрос составила 5,47 секунд. Таким образом, применяя формулу (23), итоговая оценка интерпретируемости составляет I_{LLM}=0,478.

Таблица 3 - Результаты 3 этапа эксперимента

Повтор	TP	TN	FP	FN	Время, с	P	R	F1
1	27	23	7	3	4,970	0,794	0,900	0,844
2	26	24	6	4	5,410	0,813	0,867	0,839
3	27	21	9	3	5,020	0,750	0,900	0,818
4	23	24	6	7	5,460	0,793	0,767	0,780
5	28	25	5	2	5,920	0,848	0,933	0,889
6	25	24	6	5	4,970	0,806	0,833	0,820
7	26	25	5	4	5,230	0,839	0,867	0,852
8	23	27	3	7	5,330	0,885	0,767	0,821
9	25	26	4	5	5,390	0,862	0,833	0,847
10	23	22	8	7	5,110	0,742	0,767	0,754
11	25	23	7	5	4,960	0,781	0,833	0,806
12	24	26	4	6	5,760	0,857	0,800	0,828
13	27	21	9	3	5,430	0,750	0,900	0,818
14	27	23	7	3	5,200	0,794	0,900	0,844
15	24	23	7	6	5,530	0,774	0,800	0,787
16	25	24	6	5	5,860	0,806	0,833	0,820
17	25	26	4	5	5,460	0,862	0,833	0,847
18	26	23	7	4	6,120	0,788	0,867	0,825
19	26	23	7	4	4,970	0,788	0,867	0,825
20	24	25	5	6	5,230	0,828	0,800	0,814
21	27	25	5	3	5,900	0,844	0,900	0,871
22	24	22	8	6	5,720	0,750	0,800	0,774
23	28	23	7	2	5,320	0,800	0,933	0,862
24	24	26	4	6	4,960	0,857	0,800	0,828
25	24	24	6	6	5,720	0,800	0,800	0,800
26	23	24	6	7	5,730	0,793	0,767	0,780
27	27	25	5	3	5,700	0,844	0,900	0,871
28	26	23	7	4	5,690	0,788	0,867	0,825
29	27	23	7	3	6,170	0,794	0,900	0,844
30	25	26	4	5	5,780	0,862	0,833	0,847
31	26	26	4	4	6,020	0,867	0,867	0,867
32	25	25	5	5	5,760	0,833	0,833	0,833
Ст. откл	1,273	1,262	1,262	1,273	0,311	0,033	0,042	0,023
Среднее	25,289	23,965	5,526	4,100	5,470	0,810	0,843	0,826
Медиана	25,000	24,000	6,000	5,000	5,460	0,803	0,833	0,826

Выводы по результатам экспериментального исследования

В результате анализа полученных в ходе экспериментов данных, а также результатов самого эксперимента, можно сделать следующие выводы:

1. Предложенный метод повышения интерпретации позволяет статистически значимо точность работы IDS (в среднем на 20,09% и более) путем дополнительной проверки результатов.

2. Испытуемые в абсолютном большинстве случаев повторяют ответы LLM (корреляция в ходе эксперимента составила 0,898), при этом время, требуемое для работы LLM значительно ниже, чем время ответа человека;

3. Мультиагентный подход при работе с разработанным методом расширенной интерпретации позволяет получить точность выше медианной (в сравнении с экспериментом при участии человека), в то же время принимает решение в 4,4 раза быстрее, что позволяет исключить из цепочки принятия решения человека и повысить итоговую точность системы на 23,8%. Все результаты были получены автором лично и были частично опубликованы в соавторстве (вклад автора более 80%) [106, 112].

2.5 Внедрение результатов исследования

Результаты описанного в данной главе исследования внедрены в ООО «ВойсКоммьюникэйшн», что подтверждается актом внедрения результатов (приложение 8). Использование модуля интерпретации решений нейросетевого классификатора позволяет проводить более глубокий анализ в системах прогнозирования пиковой сетевой нагрузки центра обработки данных, что в среднем повышает их эффективность на 15% за счет адекватного предиктивного масштабирования и своевременного перераспределения ресурсов.

В ООО «ЮКИТЕХ ЛАБ» применяют разработанный подход к повышению точности работы модели системы обнаружения сетевых вторжений при эксплуатации нейросетевых классификаторов, что позволяет достичь повышения точности до 12% при применении автоматической проверки данных с применением LLM. Факт внедрения подтверждается актом внедрения (приложение 8). В качестве LLM применяется GPT4.

Предлагаемых подход автоматической проверки и корректировки ответов большой языковой моделью на основе расширенной интерпретации внедрен в ООО «НТЦ «Системинвест», что также позволило повысить общую точность системы обнаружения вторжений на 12% (приложение 8).

Кроме того, основные результаты внедрены в учебный процесс Владимирского государственного университета (приложение 8).

Выводы по главе 2

1. Разработан алгоритм оценки интерпретируемости результатов работы нейросетевого анализатора трафика и алгоритм взаимодействия агентов большой языковой модели. Алгоритм позволяет численно охарактеризовать степень интерпретируемости результатов работы модели нейросети, что также позволит производить сравнение различных подходов к интерпретации.

2. Реализован модифицированный в части представления и анализа данных алгоритм интерпретации решений классификатора анализатора сетевого трафика, использующий большую языковую модель для повышения интерпретации. Реализовано API для проведения эксперимента и программа для статистической оценки результатов эксперимента. Созданы экспериментальные стенды.

Проведены эксперименты, в ходе которых определено, что при возможности обращения к большой языковой модели, испытуемые склонны соглашаться с решением последней, что происходит в абсолютном большинстве случаев (корреляция ответов испытуемых и ответов LLM составила в эксперименте 0,898). Подтверждено повышение интерпретируемости результатов модели до 0,419 (и до 0,478 в случае мультиагентного подхода), а также подтверждена гипотеза о возможности исключения человека из цепочки принятия решений на основании разработанного расширенного метода интерпретации решений нейросетевого классификатора при применении LLM и мультиагентного подхода. Исключение человека позволит, помимо прочего, сократить время обработки результатов

интерпретации до 4,4 раз при более высокой итоговой точности (0,826 против средней 0,801 F1 в ходе экспериментов). Предложенный метод интерпретации позволяет повысить итоговую точность системы более чем на 20%. Результаты экспериментов были получены автором лично и были частично опубликованы с соавторством (вклад автора более 80%) [106, 112].

3. Выполнена вторая задача исследования, а также часть четвертой задачи. Для достижения цели исследования необходимо разработать алгоритм оценки робастности нейросети и провести его экспериментальное исследование.

3 РАЗРАБОТКА АЛГОРИТМА ОЦЕНКИ ФУНКЦИОНАЛЬНОЙ УСТОЙЧИВОСТИ НЕЙРОСЕТИ АНАЛИЗАТОРА СЕТЕВОГО ТРАФИКА

В данной главе описывается ход разработки алгоритма оценки функциональной устойчивости нейросети анализатора сетевого трафика в условиях искажений входных данных. Описывается порядок и условия проведенных экспериментов и анализ результатов.

3.1 Функциональная устойчивость модели

Функциональная устойчивость, или робастность, является крайне важным показателем, характеризующим степень пригодности конкретного прибора, алгоритма или модели к реальной эксплуатации. Так, разработанный и протестированный в цеху автомобиль может фактически эксплуатироваться в условиях крайнего севера в условиях, не предусмотренных конструкторами и заводом-изготовителем, а обученная на ограниченном наборе нейросетевая модель может столкнуться с искажениями во входных данных, например, при нестабильной работе датчиков.

Рассматривая реальные сценарии использования нейросетевых классификаторов, можно разделить источники искажений входных данных на две категории: случайные искажения и злонамеренные искажения.

1. Случайные искажения: к случайным можно отнести искажения, возникающие при:
    a. Изменении распределения данных;
    b. Помехах в работе источников данных;
    c. Изменении среды эксплуатации модели.
2. Злонамеренные искажения – к данной группе относятся, прежде всего, состязательные воздействия.

Важно отметить, что проведение классических состязательных воздействий значительно затруднено ввиду природы сетевых данных. Так, например, число пакетов, используемый протокол и другие – физически получаются из сетевого оборудования, часто имеют ограниченный набор значений.

В то же время актуальной остается состязательное воздействие, заключающееся во внедрении математической закладки («отравление» выборки), однако проведение таких воздействий имеет строгую нацеленность на определенный шаблон в данных, который может быть в т. ч. типовым сценарием проведения сетевого воздействия. Такие данные не являются искаженными и не подлежат оценке в настоящем исследовании.

Рассмотрим потенциальные причины случайных искажений, являющихся актуальными для сетевого анализатора трафика.

Так, изменения распределения входных данных могут появляться вследствие изменений законодательства (переход на летнее время, изменение рабочего времени), изменение ситуации на предприятии (увольнение/прием сотрудников) и др.

Среди помех в работе источников данных можно отметить сбои оборудования в следствии перегрева (в частности, повышение времени обработки пакета и, как следствие, увеличение интервала между пакетами), временная неработоспособность оборудования в следствии аппаратных сбоев и прочие.

Кроме того, в разрезе модели изменение среды эксплуатации модели, например, подключение модели на другой сетевой контур, или значительная перестройка сетевой топологии, воспринимается как искажения во входных данных.

Для повышения функциональной устойчивости модели применяются следующие подходы:

1. Аугментация данных. Увеличение объема обучающей выборки практически всегда сильно ограничено. При этом, высокая признаковая размерность и маленький объем обучающей выборки приводит к переобучению, т.к. неспособности модели классифицировать новые данные. Применение методов

аугментации позволяет синтетическим путем увеличить объем и вариативность выборки, подготовив тем самым модель классификатора к вариативным данным промышленной среды эксплуатации [73, 113–115].

2. Использование регуляризации: применение методик регуляризации вынуждает модель изучать наиболее робастные признаки, сокращая возможность запоминания неробастных, состоящих частично из шума признаков. Такие техники, как применение drop-out, L1-L2 регуляризации, снижение сложности модели (в т. ч. при помощи технологии прунинга и дистилляции) доказанно повышают генерализацию модели [35, 116, 117].

3. Другие методы, в том числе, калибровка вероятностей, тщательный анализ обучающей выборки на предмет ошибок, вариативности, балансировка соотношения классов, специфические функции потерь так же повышают робастность моделей [118–120]. Так же контроль переобучения: обязательное использование валидационной hold-out выборки, кросс-валидация в совокупности с использованием подходящих к задаче метрик позволяет повысить функциональную устойчивость классификатора.

3.2 Алгоритм оценки функциональной устойчивости нейросети анализатора сетевого трафика в условиях искажений входных данных

Оценки функциональной устойчивости классификатора позволит характеризовать различные модели-кандидаты по степени их при годности к промышленной эксплуатации.

Разберем, что есть функциональная устойчивость нейросети.

В работе [121] дано следующее определение функциональной устойчивостью информационной системы: способность сохранения и/или восстановления данных (устойчивость) функций в условиях различного рода неблагоприятных воздействий.

Отталкиваясь от предложенной идеи, сформулируем определение функциональной устойчивости нейросети.

Определение: функциональной устойчивостью нейросети называется способность сохранять достигнутые в ходе обучения и тестирования точностные характеристики моделью при внедрении ее в промышленную эксплуатацию и связанных с этим возможных искажений в классифицируемых данных.

Поскольку в настоящем исследовании рассматривается нейросетевой анализатор сетевого трафика, а также учитывая актуальные источники искажений данных, адаптируем определение:

Определение: функциональной устойчивостью нейросети анализатора сетевого трафика называется способность сохранять достигнутые в ходе обучения и тестирования точностные характеристики моделью при внедрении ее в промышленную эксплуатацию при возможных искажениях в данных, вызванных изменением в распределении данных, помехами сетевого оборудования, изменениями в среде эксплуатации модели.

Таким образом, полученная оценка функциональной устойчивости должна отражать готовность модели к таким искажениям в данных.

Учитывая вышесказанное, в настоящем исследовании предлагается следующий алгоритм оценки функциональной устойчивости нейросети:

Алгоритм оценки глобальной робастности модели - базовый

Шаг 1. Получить тестовую выборку размера не менее M элементов, не использованную при обучении классификатора.

Шаг 2. Выбрать подходящий под тип данных метод аугментации.

Шаг 3. Произвести классификацию тестовых данных и оценить точность классификации по выбранной метрике S.

Шаг 4. Для каждого элемента тестовой выборки сформировать N аугментированных примеров с выбранной силой аугментации - магнитудой (при возможности).

Шаг 5. Провести классификацию полученных на шаге 4 данных (без учета исходных данных) и оценить точность классификации по выбранной метрике S.

Шаг 6. Произвести расчет оценки робастности для модели.

Конец.

Значение для N выбирается 1 и более. Значение для M обычно равно размеру валидационной или тестовой выборки.

При создании искажений в данных для оценки робастности важно учитывать природу данных. Так, недопустимо, чтобы генератор шума создавал примеры с вещественным числом пакетов, либо использовал несовместимые с определенным протоколом флаги. Учитывая это, в настоящей работе предлагается в качестве генератора зашумленных синтетических данных использовать cVAE - вариационный автокодировщик с условием. Итоговый алгоритм, используемый в настоящей работе:

$$R = \frac{\min(S, \hat{s})}{S} * \left(1 - R_{overfit}\right) \quad (25)$$

где где S – исходная точность, $\hat{s}$ - точность на аугментированных данных.

$$R_{overfit} = \frac{\max(0; F_{train} - F_{test})}{F_{train}} \quad (26)$$

где F – точность на обучающем или тестовом наборе.

В экспериментах используется метрика F1-score. Для аугментации данных в настоящем исследовании предлагается использовать следующий алгоритм.

Алгоритм аугментации данных

Шаг 1. Обучение модели cVAE высокой точности

Шаг 2. Генерация N данных:

1. Кодирование данных для получения векторов среднего (mu) и логарифма дисперсии, к которому добавляется шум из z-распределения с выбранной магнитудой. Категориальные данные, включая метку класса – используются как условия.
2. Вектор z сэмплируется из нормального распределения с параметрами mu и std, при помощи повторной параметризации
3. Денормализация данных.
4. Расчет зависимых признаков, округление с учетом типов данных.
5. Нормализация данных.

Конец.

При применении алгоритма аугментации данных важно учитывать исходную природу признаков. Так, в аугментированных данных не должно получаться дробное число принятых/отправленных пакетов, отрицательная длительность сессии, несуществующих в протоколе флагов пакетов (например, флаг ACK при UPD-соединении, либо одновременная комбинация CYN и FIN флагов). Кроме того, часть признаков в исследуемом наборе данных является рассчитанными, или зависимыми – генерировать их значения моделью не только избыточно, но и может приводить к нестыковкам.

С учетом вышесказанного, в работе были разработаны требуемые алгоритмом функции:

4. Денормализация - позволяет оперировать значениями признаков в исходных величинах для обеспечения рациональности значений независимых признаков;
5. Расчет зависимых признаков – функционал для получения зависимых признаков на основе независимых;
6. Нормализация – возвращает значения признаков к требуемому моделью нейросети масштабу.

3.3 Разработка алгоритма оценки адаптируемости

В настоящем диссертационном исследовании, адаптируемостью называется свойство системы, использующей нейросетевой классификатор, определять момента наступления потребности в переобучении модели путем мониторинга классифицируемых данных, ответов модели, их распределения.

Для упрощения оценки, в случае наличия настроенной системы мониторинга, оценка адаптируемости принимает значения 1.

В противном случае, необходимо оценить вариативность распределений обучающей и тестовой выборок. Чем более постоянны распределения признаков, тем меньше вероятность того, что при эксплуатации модель будет сталкиваться с

данными, существенно отличающимися от тех, на которых она была обучена, т. е. фактор смены распределения будет иметь низкую значимость.

Оценка вариативности распределений можно производить с использованием статистических тестов, таких как тест Колмогорова-Смирнова для непрерывных признаков и критерий Хи-квадрат для категориальных признаков. Эти тесты позволяют количественно оценить различия в распределениях данных, что служит основой для определения адаптируемости модели.

В роботе предлагается следующий алгоритм оценки адаптируемости. Программная реализация алгоритма представления в приложении 4.

Алгоритм оценки адаптируемости

Шаг 1. Разделить выборку на тренировочную и тестовую по времени в соотношении 70/30*.

Шаг 2. Для каждого признака из обучающего набора оценить изменение в распределении при помощи теста Колмогорова-Смирнова**.

Шаг 3. Для полученных значений рассчитать $\mathrm{p} = \sup\left(\left[1 - \mathrm{p}_{\mathrm{value_i}}\right]\right)$ где i обозначает i-й признак.

Шаг 4. Оценить адаптируемость модели по формуле:

$$A = \max\left(\hat{A}, \frac{-\log(p+e)}{-\log(e)}\right) \quad (27)$$

где $\hat{\mathrm{A}}$ - флаг-индикатор наличия системы автомониторинга.

* При невозможности подобного разбиения – применяется разбиение на обучающую и тестовую выборки

** Для категориальных признаков используется критерий Хи-квадрат

3.4 Стенд проведения эксперимента

Для проведения эксперимента был подготовлен стенд с ЦПУ AMD Ryzen 7950X, оперативной памятью 64GB DDR5, графическим ускорителем NVIDIA

RTX 3090.

Для возможности повторного получения аналогичных результатов все начальные значения генераторов случайных чисел были зафиксированы. В качестве языка программирования использовался Python версии 3.10, фреймворк для глубокого обучения и эксплуатации модели Pytorch 2.0.0+cuda.

Кроме того, были задействованы следующие библиотеки:

– Pandas: использовалась для загрузки, очистки и предобработки данных, а также для анализа данных и манипулирования таблицами данных.

– Numpy: применялась для выполнения математических и логических операций на массивах, а также для обработки многомерных массивов данных.

– Matplotlib: использовалась для визуализации данных и результатов моделирования, распределения данных.

– Mlflow: применялась для отслеживания экспериментов, записи и управления результатами.

– Pickle: использовалась для сериализации и десериализации объектов Python, т. е. при сохранении и загрузке моделей и других данных.

– DVC (Data Version Control): применялась для управления версиями данных и моделей, а также для воспроизводимости экспериментов и пайплайнов обработки данных.

3.5 Экспериментальная проверка предложенного алгоритма

Для проверки разработанного алгоритма требуется определить подходы, доказанно повышающие робастность нейросетевой модели. Среди них: увеличение объема обучающей выборки, использование регуляризации, дропаут, пакетная нормализация, балансировка классов.

Предполагается, добавление в исходную (базовую) архитектуру модели перечисленных выше методов, адекватная оценка робастности должна статистически значимо повышаться. Кроме того, ожидается, что модели с более

высокой оценкой робастности будут демонстрировать значимо меньшее падение точности на состязательных данных.

Для получения аугментированных данных (приложение 6) была обучена модель cVAE, содержащая 9 слоев и использующая комбинированную функцию потерь: среднеквадратичную ошибку и дивергенцию Кульбака-Лейблера. Обученная модель демонстрирует значение функции потерь, равное 8,42.

Параметры подобранной и обученной в ходе экспериментов архитектуры представлены на рис. 21. Скорость обучения была выбрана 1e-5, размер пакета 51200 примеров. В качестве критерия остановки обучения выбран критерий ранней остановки с терпимостью 9 эпох. Наилучшая эпоха по результатам обучения – 470.

В качестве базовой архитектуры выступает полносвязная многослойная модель прямого распространения с 4 слоями, параметры слоев которой подобраны аналогично описанному в главе 2 эксперименту.

В экспериментах применялись выборки из исходного набора AWS IDS-2018. Модели обучались на наборе 1000000 строк, тестировались на валидационном и тестовом наборах размером 600000 строк. В ходе экспериментов в статистических тестах использовался уровень значимости 0,05.

Layer (type:depth-idx)	Output Shape	Param #
CAutoencoder	[1, 24]	48
├─Linear: 1-1	[1, 256]	10,496
├─BatchNorm1d: 1-2	[1, 256]	512
├─ReLU: 1-3	[1, 256]	--
├─Dropout: 1-4	[1, 256]	--
├─Linear: 1-5	[1, 128]	32,896
├─BatchNorm1d: 1-6	[1, 128]	256
├─ReLU: 1-7	[1, 128]	--
├─Dropout: 1-8	[1, 128]	--
├─Linear: 1-9	[1, 128]	16,512
├─BatchNorm1d: 1-10	[1, 128]	256
├─ReLU: 1-11	[1, 128]	--
├─Dropout: 1-12	[1, 128]	--
├─Linear: 1-13	[1, 10]	1,290
├─BatchNorm1d: 1-14	[1, 10]	20
├─Linear: 1-15	[1, 10]	110
├─Linear: 1-16	[1, 10]	110
├─Linear: 1-17	[1, 10]	270
├─BatchNorm1d: 1-18	[1, 10]	20
├─ReLU: 1-19	[1, 10]	--
├─Dropout: 1-20	[1, 10]	--
├─Linear: 1-21	[1, 128]	1,408
├─BatchNorm1d: 1-22	[1, 128]	256
├─ReLU: 1-23	[1, 128]	--
├─Dropout: 1-24	[1, 128]	--
├─Linear: 1-25	[1, 128]	16,512
├─BatchNorm1d: 1-26	[1, 128]	256
├─ReLU: 1-27	[1, 128]	--
├─Dropout: 1-28	[1, 128]	--
├─Linear: 1-29	[1, 256]	33,024
├─BatchNorm1d: 1-30	[1, 256]	512
├─ReLU: 1-31	[1, 256]	--
├─Dropout: 1-32	[1, 256]	--
├─Linear: 1-33	[1, 24]	6,168

Рисунок 21 - Архитектура используемой модели cVAE

Всего было проведено 3 эксперимента, в ходе которых было обучено 153 модели, разбитых на группы:

1. группа состоит из 32 базовых модели с различными комбинациями методов повышения робастности;

2. группа состоит из 57 базовых моделей, отличающиеся друг от друга используемой инициализацией генератора случайных значений.

Каждая модель обучалась с использованием оптимизатора функции потерь Adam. В Качестве функции потерь была выбрана перекрестная энтропия. Для

оптимизации времени обучения использовался критерий ранней остановки с терпимостью 4 эпохи. Скорость обучения составляла 1e-3.

Дополнительно было обучено 64 моделей: по 32 модели с различным числом и шириной слоев, с различными комбинациями методов повышения робастности (группа 1) и без методов повышение робастности (группа 2).

Для каждой модели рассчитывалась оценка робастности R, оценивалось падение уверенности в предсказании на состязательных данных и число шагов до смены предсказания на состязательных данных.

На рис. 22 представлены результаты сравнения R в группах.

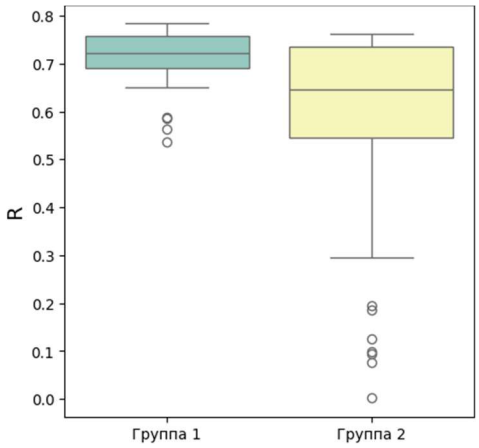


Рисунок 22 - Сравнение R в группах, одна архитектура

Для проверки различия дисперсий применялся тест Левана, показавший их значимое отличие ($p = 0{,}0027$). Для проверки различия средних в данном случае рекомендовано применение теста Манна-Уитни. Односторонний тест Манна-Уитни показал значимо большее значение R в группе 1 ($p = 0{,}00068$). Интересным заключением является наличие среди группы 2 моделей с высокой оценкой робастности R.

Еще более выраженное отличие было получено при использовании моделей с различной архитектурой (рис. 23). Тест Левена показал значимое отличие дисперсий ($p = 0{,}046$), потому для проверки разницы средних применялся односторонний тест Манна-Уитни. В таком случае, статистически значимо значение R больше в группе 1 ($p = 5{,}26 \times 10^{-6}$).

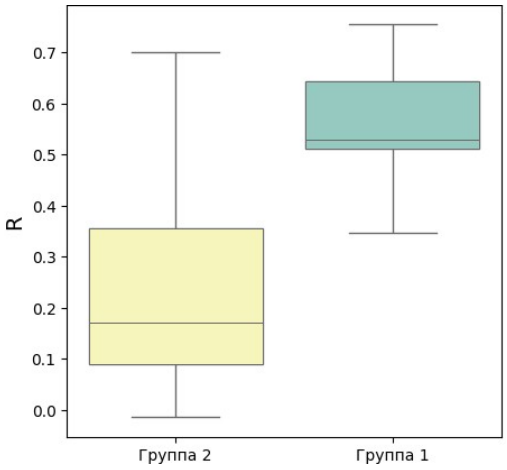


Рисунок 23 - Сравнение R в группах, разные архитектуры

Следующий эксперимент проверял корреляцию между значением падения уверенности в предсказаниях на состязательных данных и оценкой робастности.

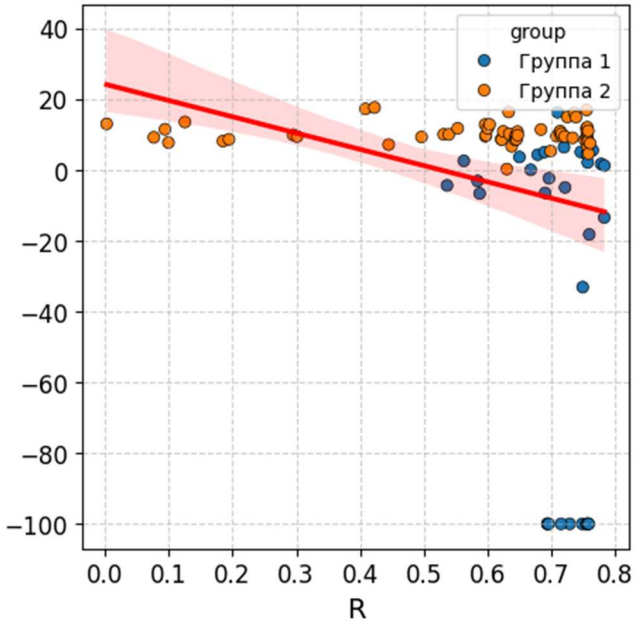


Рисунок 24 - Снижение уверенности в предсказании, одна архитектура

Для генерации состязательных данных применялся Быстрый Метод Знака Градиента (FGSM) [122] с epsilon=0,04 (приложение 5). На рис. 24 и рис. 25 показаны графики снижения уверенности в предсказании на состязательных данных, с нанесенной линией тренда, отражающей отрицательную корреляцию, при использовании одной архитектуры и различных архитектур, соответственно.

В случае одной архитектуры, ввиду ненормальности данных в выборках ($p = 1{,}6 \times 10^{-10}$ и $2{,}1 \times 10^{-12}$), была рассчитана корреляция Спирмена, составившая -0,288, что является значимым ($p = 0{,}005$) с выбранным уровнем значимости.

При использовании различных архитектур (рис. 25) коэффициент корреляции Спирмена повышается и становится -0,704, что так же является значимым ($p = 8{,}46 \times 10^{-1}$). Данные также не являются нормально распределенными ($p = 9{,}75 \times 10^{-13}$ и 0,00025).

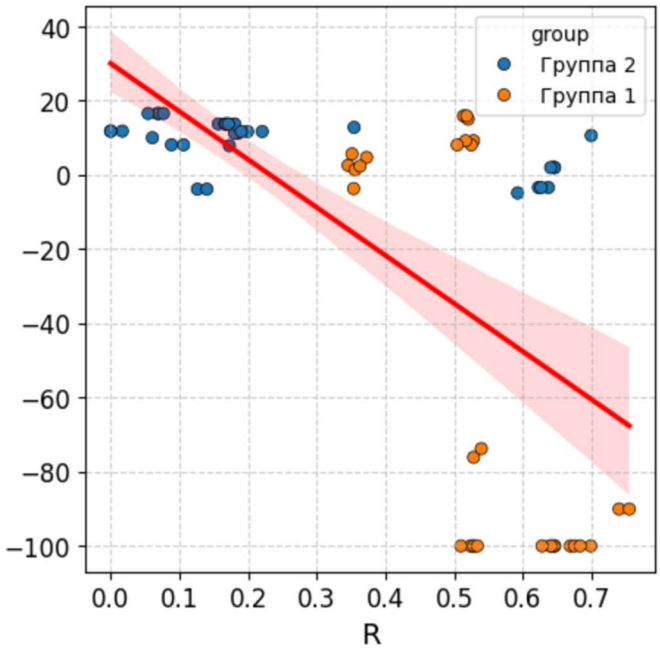


Рисунок 25 - Снижение уверенности в предсказании, разные архитектуры

В следующем эксперименте проверялась корреляция между требуемым количеством шагов для смены предсказания на противоположное. При этом применялся итеративный метод генерации состязательных данных – DeepFool [123] с шагом 0,02 (приложение 7).

На рис. 26 представлены результаты эксперимента при использовании одной архитектуры. Данные в выборках не являются нормально распределенными ($p = 1{,}68 \times 10^{-10}$ и $2{,}5 \times 10^{-1}$). Корреляция Спирмена составила 0,518, что является статистически значимым ($p = 1{,}4 \times 10^{-7}$) с выбранным уровнем значимости.

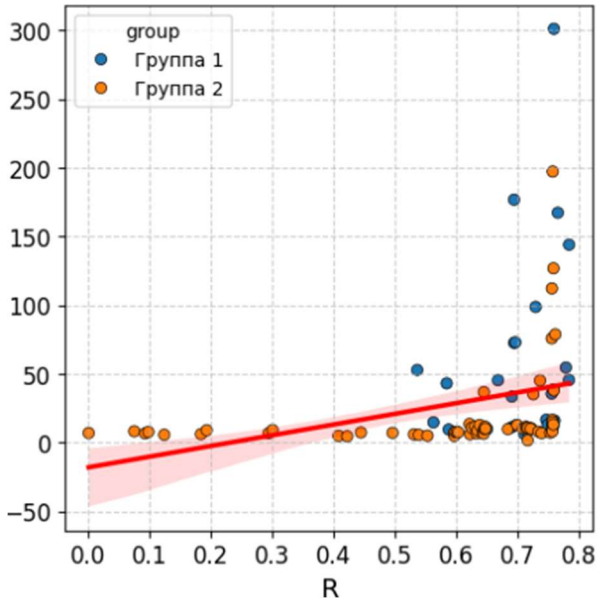


Рисунок 26 - Число шагов до смены предсказания, одна архитектура

На рис. 27 представлены результаты эксперимента при использовании различных архитектур архитектуры. Данные в выборках не являются нормально

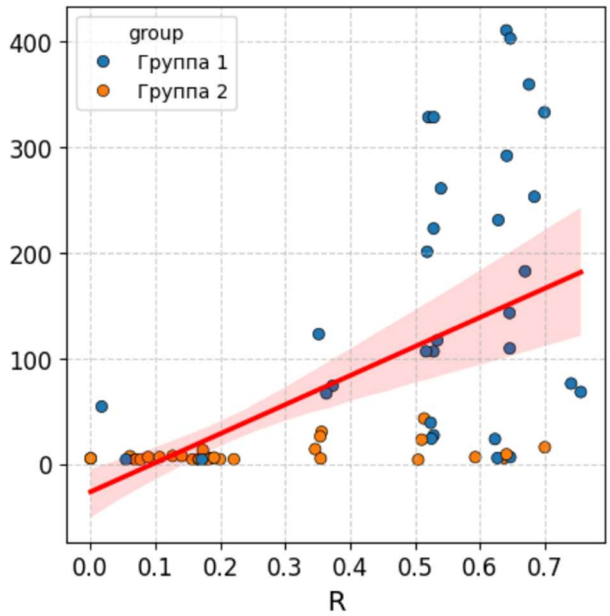


Рисунок 27 - Число шагов до смены предсказания, разные архитектуры

распределенными ($p = 9{,}7 \times 10^{-1}$ и $4{,}2 \times 10^{-5}$). Корреляция Спирмена в данном случае составила 0,674, что является статистически значимым ($p = 1{,}05 \times 10^{-9}$) с выбранным уровнем значимости.

Выводы по результатам экспериментального исследования

В результате проведения экспериментов можно сделать следующие выводы:

1. Значение оценки, полученное при помощи предложенного алгоритма оценки функциональной устойчивости нейросети анализатора сетевого трафика, имеет статистически значимую корреляцию с падением уверенности в предсказании – чем выше оценка, тем меньше падение уверенности и больше шагов требуется при генерации примеров для проведения успешного состязательного воздействия. Это соответствует выдвинутым перед началом эксперимента предложениям и свидетельствует об адекватности предложенной оценки.

2. Среднее значение оценки функциональной устойчивости нейросети в группах с методами, повышающими робастность и без значимо отличается в пользу группы с методами, повышающими робастность, что также подтверждает адекватность алгоритма.

Предложенный алгоритм стабильно работает не зависимо от архитектуры. При этом, при использовании различных архитектур разница средних и корреляция более выражена, что может свидетельствовать о зависимости восприимчивости к состязательным воздействиям от конкретной архитектуры модели. Результаты получены автором лично и частично опубликованы в соавторстве (вклад автора более 90%) [107].

3.6 Расчет повышения качества классификатора

Таким образом, в ходе экспериментов подтвердилась адекватность предполагаемой формулы оценки функциональной устойчивости нейросети анализатора сетевого трафика.

Оценим повышение качества классификатора, достигнутое в ходе настоящего исследования. Результаты расчетов представлены в табл. 4.

В качестве *Базовой модели 1* выступает отобранная перебором параметров в ходе экспериментов полносвязная модель с 4 слоями с размерностями 45-128-512-128-2, пакетной нормализацией после каждого скрытого слоя, имеющая точность на тестовой выборке F1-score 0,9875. Оценка робастности (R) данной модели составляет 0,5105.

Базовая модель 2– та же базовая модель 1, но при эксплуатации которой применяются системы автомониторинга.

Базовая модель 3 представляет собой случай, когда была случайно выбрана наиболее робастная модель, а также применяется система автомониторинга.

Конечной моделью является наиболее робастная модель из моделей с высокой точностью, при использовании которой применяется разработанный алгоритм повышения интерпретации решений, а также применяется мониторинг потребности в адаптации.

Для расчета оценки адаптируемости в базовом случае был разработан код (приложение 7), использующий формулу 5. Супремум $1 - p$-значение статистик Колмагорова-Смирнова для непрерывных и Хи-квадрат составил 0,9826, оценка адаптируемости в базовом случае - 0,00162.

Таблица 4. Расчет качества классификаторов

	Базовая 1	Базовая 2	Базовая 3	Конечная	Конечная LLM	Конечная $LLM_{м.аг.}$
C	Да	Да	Да	Да	Да	Да
F	0,988	0,988	0,984	0,984	0,984	0,984
I	Нет	Нет	Нет	0,403	0,419	0,478
R	0,511	0,511	0,758	0,758	0,758	0,758
A	0,002	Есть сист. автомон.	Есть сист. автомон.	Есть сист. автомон.	Есть сист. автомон.	Есть сист. автомон.
P	0,169	0,497	0,577	0,709	0,714	0,733

Таким образом, в ходе работы удалось повысить качество модели в 4,2 раза и 4,4 относительно базовой модели 1, на 43% и 48% относительно базовой модели 2 и на 23% и 27% относительно базовой модели 3 при использовании людей в экспериментах и с применением только большой языковой модели с мультиагентным подходом соответственно. Результаты получены автором лично и частично опубликованы [110].

3.7 Внедрение результатов исследования

Результаты описанного в данной главе исследования в виде алгоритма оценки функциональной устойчивости нейросети анализатора сетевого трафика

внедрены в ООО «АйТиАрт» г. Москва, что подтверждается актом внедрения результатов кандидатской диссертационной работы (приложение 8).

Использование данного алгоритма позволило снизить деградацию точности со временем модели нейросетевого анализатора сетевых вторжений на 10% в среднем. Кроме того, благодаря возможности отбора наиболее робастной модели, а также расширении использования методов регуляризации, время до возникновения потребности в переобучении удалось увеличить до 20% по сравнению с базовыми моделями.

Внедрение в учебный процесс Владимирского государственного университета поспособствовало улучшению качества учебного процесса (приложение 8).

Выводы к главе 3

1. Дано определение функциональной устойчивости нейросети. Функциональная устойчивость (робастность) отражает способность модели сохранять характеристику точности при искажениях во входных данных. Определены источники искажений во входных данных: случайные и злонамеренные. Наиболее актуальными источниками искажений для нейросетевых анализаторов трафика являются случайные изменения, появляющиеся ввиду изменений условий эксплуатации модели.

2. Разработан алгоритм оценки функциональной устойчивости нейросети анализатора сетевого трафика, позволяющий численно охарактеризовать способность конкретной модели сохранять стабильность характеристики точности при возможной вариативности входных данных, в т. ч. позволяющий сравнивать модели-кандидаты между собой. Алгоритм стабильно работает не зависимо от архитектуры. Предложен алгоритм аугментации данных для использования при оценке функциональной устойчивости, применяющий модель вариационного автокодировщика с условием и учитывающий природу данных.

3. Создан экспериментальный стенд и проведены эксперименты, подтверждающие адекватность предложенного алгоритма оценки функциональной устойчивости нейросети. Модели, обученные с применением методов повышения робастности, имеют статистически значимое отличие средних оценок устойчивости R. Модели с высокой оценкой устойчивости демонстрируют значимо меньшее снижение уверенности предсказаний на состязательных данных, требуют больше итераций для проведения успешного воздействия, т. е. являются более подготовленными к возможным зашумлениям в данных.

4. Выполнена третья и четвертая задачи исследования. Разработанный алгоритм оценки устойчивости работает стабильно на различных архитектурах нейросетей прямого распространения, что подтверждено в ходе экспериментов. Достигнута цель работы. Разработанные в ходе настоящего исследования модели и алгоритмы позволяют повысить качество модели IDS, непосредственно влияющее на эффективность принятия решений, в 4,2 раза и 4,35 относительно базовой модели 1, на 43% и 48% относительно базовой модели 2 и на 23% и 27% относительно базовой модели 3 при использовании людей в экспериментах и с применением только большой языковой модели с мультиагентным подходом соответственно. Результаты получены автором лично и частично опубликованы [107, 110].

ЗАКЛЮЧЕНИЕ

В результате выполнения данного диссертационного исследования можно сделать следующие теоретические и практические выводы:

– В результате анализа нейронных сетей определены особенности данного алгоритма машинного обучения, которые необходимо учитывать при использовании: отсутствие интерпретации решений, низкая робастность, восприимчивость к состязательным воздействиям, потребность в адаптации и определении момента необходимости проведения адаптации. Несмотря на широкое применение нейросетей, в настоящее время уделяется недостаточное внимание данным аспектам, что приводит к снижению доверия и функциональной устойчивости алгоритма. В то же время не существует единой оценки интерпретируемости и робастности нейросетей. Результаты анализа проблем состязательных воздействий, робастности, интерпретации, мониторинга и адаптации нейросетей, определение качества классификатора и его составляющих получены автором лично и были частично опубликованы, в т. ч. в соавторстве (вклад автора более 80%) в [79, 106–109].

– Выполнена первая задача исследования. Изучены основные сферы применения нейросетей в задачах, связанных с анализом сетевого трафика – это системы обнаружения, предотвращения вторжений; системы мониторинга производительности сети; системы глубокого пакетного анализа трафика, системы управления сетевым трафиком. В работе более детально исследуется применение нейросетевых классификаторов в системах обнаружения сетевых вторжений.

– Уточнено понятие качества нейросетевого классификатора и выделены составляющие качества нейросетевого классификатора; определено влияние качества модели на эффективность принятия решений. Сформулирована задача оптимизации, заключающаяся в повышении качества нейросетевого классификатора в условиях ограниченности времени получения информации от СППР и отсутствии возможности обучения специалистов, потенциально

использующих систему, машинному обучению или иным дополнительным сферам. Повышение качества классификатора можно достичь через реализацию расширенной интерпретации решений модели, а также разработку алгоритм оценки функциональной устойчивости нейросети для выбора наиболее робастной модели.

– Разработан алгоритм оценки интерпретируемости результатов работы нейросетевого анализатора трафика и алгоритм взаимодействия агентов большой языковой модели. Алгоритм позволяет численно охарактеризовать степень интерпретируемости результатов работы модели нейросети, что также позволит производить сравнение различных подходов к интерпретации.

– Реализован модифицированный в части представления и анализа данных алгоритм интерпретации решений классификатора анализатора сетевого трафика, использующий большую языковую модель для повышения интерпретации. Реализовано API для проведения эксперимента и программа для статистической оценки результатов эксперимента. Созданы экспериментальные стенды.

– Проведены эксперименты, в ходе которых определено, что при возможности обращения к большой языковой модели, испытуемые склонны соглашаться с решением последней, что происходит в абсолютном большинстве случаев (корреляция ответов испытуемых и ответов LLM составила в эксперименте 0,898). Подтверждено повышение интерпретируемости результатов модели до 0,419 (и до 0,478 в случае мультиагентного подхода), а также подтверждена гипотеза о возможности исключения человека из цепочки принятия решений на основании разработанного расширенного метода интерпретации решений нейросетевого классификатора при применении LLM и мультиагентного подхода. Исключение человека позволит, помимо прочего, сократить время обработки результатов интерпретации до 4,4 раз при более высокой итоговой точности (0,826 против средней 0,801 F1 в ходе экспериментов). Предложенный метод интерпретации позволяет повысить итоговую точность системы более чем на 20%.

– Выполнена вторая задача исследования, а также часть четвертой задачи. Для достижения цели исследования необходимо разработать алгоритм оценки робастности нейросети и провести его экспериментальное исследование.

– Дано определение функциональной устойчивости нейросети. Функциональная устойчивость (робастность) отражает способность модели сохранять характеристику точности при искажениях во входных данных. Определены источники искажений во входных данных: случайные и злонамеренные. Наиболее актуальными источниками искажений для нейросетевых анализаторов трафика являются случайные изменения, появляющиеся ввиду изменений условий эксплуатации модели.

– Разработан алгоритм оценки функциональной устойчивости нейросети анализатора сетевого трафика, позволяющий численно охарактеризовать способность конкретной модели сохранять стабильность характеристики точности при возможной вариативности входных данных, в т. ч. позволяющий сравнивать модели-кандидаты между собой. Алгоритм стабильно работает не зависимо от архитектуры. Предложен алгоритм аугментации данных для использования при оценке функциональной устойчивости, применяющий модель вариационного автокодировщика с условием и учитывающий природу данных.

– Создан экспериментальный стенд и проведены эксперименты, подтверждающие адекватность предложенного алгоритма оценки функциональной устойчивости нейросети. Модели, обученные с применением методов повышения робастности, имеют статистически значимое отличие средних оценок устойчивости R. Модели с высокой оценкой устойчивости демонстрируют значимо меньшее снижение уверенности предсказаний на состязательных данных, требуют больше итераций для проведения успешного воздействия, т. е. являются более подготовленными к возможным зашумлениям в данных.

– Выполнена третья и четвертая задачи исследования. Разработанный алгоритм оценки устойчивости работает стабильно на различных архитектурах нейросетей прямого распространения, что подтверждено в ходе экспериментов. Достигнута цель работы. Разработанные в ходе настоящего исследования модели и

алгоритмы позволяют повысить качество модели IDS, непосредственно влияющее на эффективность принятия решений, в 4,2 раза и 4,4 относительно базовой модели 1, на 43% и 48% относительно базовой модели 2 и на 23% и 27% относительно базовой модели 3 при использовании людей в экспериментах и с применением только большой языковой модели с мультиагентным подходом соответственно. Все результаты получены автором лично и частично опубликованы [79, 106, 107, 110, 112].

В рамках дальнейших исследований планируется доработка процесса обучения классификаторов трафика в части включения проверки робастности в процесс, что позволит производить отбор оптимальной модели автоматически. Кроме того, будет продолжено исследование использования LLM для повышения точности моделей.

СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ

– ЭВМ – электронно-вычислительная машина

– ПО – программное обеспечение

– СППР – системы поддержки принятия решений

– ЦПУ – центральное процессорное устройство

– NIDS (англ. Network Intrusion Detection System) – система обнаружения сетевых вторжений

– LLM (англ. Large Language Model) – большая языковая модель

– IDS (англ. Intrusion Detection System) – система обнаружения вторжений

– IPS (англ. Intrusion Prevention System) – система предотвращения вторжений

– DPI (англ. Deep Packet Inspection) – глубокая проверка пакетов

– NPM (англ. Network Performance Monitoring) – мониторинг производительности сети

– SIPS (англ. Stateful Intrusion Prevention System) – система предотвращения вторжений с отслеживанием состояния

– AIPS (англ. Advanced Intrusion Prevention System) – расширенная система предотвращения вторжений

– AIDS (англ. Advanced Intrusion Detection System) – расширенная система обнаружения вторжений

– SDN (англ. Software-Defined Networking) – программно-определяемая сеть

– IoT (англ. Internet of Things) – интернет вещей

– CNN (англ. Convolutional Neural Network) – сверточная нейронная сеть

– RNN (англ. Recurrent Neural Network) – рекуррентная нейронная сеть

– LSTM (англ. Long Short-Term Memory) – долгосрочная кратковременная память

– GAN (англ. Generative Adversarial Network) – генеративная состязательная сеть

– DNN (англ. Deep Neural Network) – глубокая нейронная сеть

– ANN (англ. Artificial Neural Network) – искусственная нейронная сеть

– SAE (англ. Sparse Autoencoder) – разреженный автокодировщик

– cVAE (англ. Conditional Variational Autoencoder) – вариационный автокодировщик с условием

– SRT (англ. Server Response Time) – время ответа сервера

– RTT (англ. Round-Trip Time) – время прохождения туда и обратно

– WANN (англ. Weight Agnostic Neural Network) – параметро-независимая нейронная сеть

– ReLU (англ. Rectified Linear Unit) – функция активации «прямоугольная линейная единица»

– GRU (англ. Gated Recurrent Unit) – управляемая рекуррентная единица

– PDP (англ. Partial Dependence Plot) – график частичной зависимости

– LIME (англ. Local Interpretable Model-Agnostic Explanations) – локальные интерпретируемые объяснения для моделей

– SHAP (англ. SHapley Additive exPlanations) – добавочные объяснения по методу Шепли

– DOS (англ. Denial of Service) – отказ в обслуживании

– API (англ. Application Programming Interface) – программный интерфейс приложения

– DVC (англ. Data Version Control) – контроль версий данных

– TCP (англ. Transmission Control Protocol) – протокол управления передачей

– GPT (англ. Generative Pretrained Transformer) – генеративный предварительно обученный трансформер

– TP (англ. True Positive) – истинно положительное

– TN (англ. True Negative) – истинно отрицательное

– FP (англ. False Positive) – ложно положительное

– FN (англ. False Negative) – ложно отрицательное

– FGSM (англ. Fast Gradient Sign Method) – метод быстрого знака градиента

– ICS (англ. Industrial Control System) – промышленная система управления

СПИСОК ЛИТЕРАТУРЫ

1. What is Network Traffic Analysis (NTA)? – URL: https://www.rapid7.com/fundamentals/network-traffic-analysis/ (date accessed: 03.01.2024). – Text : electronic.

2. Что такое анализ сетевого трафика или как найти потенциальный канал утечки информации? – URL: https://cyberden.pw/chto-takoe-analiz-setevogo-trafika (дата обращения: 03.01.2025). – Текст : электронный.

3. IDS (система обнаружения вторжений). – URL: https://encyclopedia.kaspersky.ru/glossary/ids-intrusion-detection-system/ (дата обращения: 03.01.2025). – Текст : электронный.

4. IPS (система предотвращения вторжений). – URL: https://encyclopedia.kaspersky.ru/glossary/ips-intrusion-prevention-system/ (дата обращения: 03.01.2025). – Текст : электронный.

5. Network intrusion detection system: A systematic study of machine learning and deep learning approaches / Z. Ahmad, A. Shahid Khan, C. Wai Shiang et al. // Transactions on Emerging Telecommunications Technologies. – 2021. – Vol. 32. – Network intrusion detection system. – № 1.

6. Поздняк, И. С. Использование алгоритмов машинного обучения для обнаружения аномального поведения трафика / Поздняк И. С., Макаров И. С. // Инфокоммуникационные технологии. – 2023. – Т. 21. – № 3. – С. 20-27.

7. Альбертовна, Г. Д. АНАЛИЗ МОДЕЛЕЙ ГЛУБОКОГО ОБУЧЕНИЯ ДЛЯ ЗАДАЧ ОБНАРУЖЕНИЯ СЕТЕВЫХ АНОМАЛИЙ ИНТЕРНЕТА ВЕЩЕЙ / Г. Д. Альбертовна, К. И. Витальевич // Информационно-управляющие системы. – 2021. – № 1 (110). – С. 28-37.

8. Татарникова, Т. ОБНАРУЖЕНИЕ АТАК В СЕТЯХ ИНТЕРНЕТА ВЕЩЕЙ МЕТОДАМИ МАШИННОГО ОБУЧЕНИЯ / Т. Татарникова, П. Богданов // Информационно-управляющие системы. – 2021. – № 6 (115). – С. 42-52.

9. M. R., G. R. Machine learning for intrusion detection in industrial control systems: challenges and lessons from experimental evaluation / G. R. M. R., C. M. Ahmed, A. Mathur // Cybersecurity. – 2021. – Vol. 4. – Machine learning for intrusion detection in industrial control systems. – № 1. – P. 27.

10. MAFSIDS: a reinforcement learning-based intrusion detection model for multi-agent feature selection networks / K. Ren, Y. Zeng, Y. Zhong et al. // Journal of Big Data. – 2023. – Vol. 10. – MAFSIDS. – № 1. – P. 137.

11. CNN-IDS: Convolutional Neural Network for Network Intrusion Detection System / A. H. Halbouni, T. S. Gunawan, M. Halbouni et al. – Text : electronic // 2022 8th International Conference on Wireless and Telematics (ICWT) 2022 8th International Conference on Wireless and Telematics (ICWT). – 2022. – CNN-IDS. – P. 1-4. – URL: https://ieeexplore.ieee.org/document/9935478 (date accessed: 04.01.2025).

12. Intrusion detection systems for software-defined networks: a comprehensive study on machine learning-based techniques / Z. Mustafa, R. Amin, H. Aldabbas, N. Ahmed // Cluster Computing. – 2024. – Vol. 27. – Intrusion detection systems for software-defined networks. – № 7. – P. 9635-9661.

13. A detailed analysis of the KDD CUP 99 data set / M. Tavallaee, E. Bagheri, W. Lu, A. A. Ghorbani. – Text : electronic // 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA). – Ottawa, ON, Canada : IEEE, 2009. – P. 1-6.

14. A Survey of Methods for Explaining Black Box Models / R. Guidotti, A. Monreale, S. Ruggieri et al. // ACM Comput. Surv. – 2018. – Vol. 51. – № 5. – P. 93:1-93:42.

15. Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection / J. Kim, J. Kim, H. L. Thi Thu, H. Kim. – Text : electronic // 2016 International Conference on Platform Technology and Service (PlatCon) 2016 International Conference on Platform Technology and Service (PlatCon). – 2016. – P. 1-5. – URL: https://ieeexplore.ieee.org/abstract/document/7456805 (date accessed: 04.01.2025).

16. Алексеевич, С. Д. ПОИСК АНОМАЛИЙ С ПОМОЩЬЮ АВТОЭНКОДЕРОВ / С. Д. Алексеевич, К. Ю. Дмитриевич, З. К. Сергеевич // International Journal of Open Information Technologies. – 2022. – Т. 10. – № 8. – С. 39-45.

17. Bhatt, R. Detecting the undetectable: GAN-based strategies for network intrusion detection / R. Bhatt, G. Indra // International Journal of Information Technology. – 2024. – Vol. 16. – Detecting the undetectable. – № 8. – P. 5231-5237.

18. A Comparative Analysis of the TDCGAN Model for Data Balancing and Intrusion Detection / M. Jamoos, A. M. Mora, M. AlKhanafseh, O. Surakhi // Signals. – 2024. – Vol. 5. – № 3. – P. 580-596.

19. Albin, E. A Realistic Experimental Comparison of the Suricata and Snort Intrusion-Detection Systems / E. Albin, N. C. Rowe. – Text : electronic // 2012 26th International Conference on Advanced Information Networking and Applications Workshops 2012 IEEE Workshops of International Conference on Advanced Information Networking and Applications (WAINA). – Fukuoka, Japan : IEEE, 2012. – P. 122-127.

20. Performance comparison of intrusion detection systems and application of machine learning to Snort system // Future Generation Computer Systems. – 2018. – Vol. 80. – P. 157-170.

21. Что такое Network Performance Monitoring (NPM)? – URL: https://networkguru.ru/chto_takoe_network_performance_monitoring/ (дата обращения: 04.01.2025). – Текст : электронный.

22. Deepak, K. S. An Ultimate Guide to Network Performance Monitoring in 2025 | Infraon / K. S. Deepak. – 2022. – URL: https://infraon.io/blog/an-ultimate-guide-to-network-performance-monitoring-npm/ (date accessed: 04.01.2025). – Text : electronic.

23. Cairoli, F. Neural Predictive Monitoring under Partial Observability / F. Cairoli, L. Bortolussi, N. Paoletti arXiv:2108.07134 [cs]. – arXiv, 2021. – URL: http://arxiv.org/abs/2108.07134 (date accessed: 04.01.2025). – Text : electronic.

24. Flowmon - Trusted solution for network and security operations | Flowmon. – URL: https://www.progress.com/flowmon (date accessed: 04.01.2025). – Text : electronic.

25. Deep Packet: A Novel Approach For Encrypted Traffic Classification Using Deep Learning. Deep Packet / M. Lotfollahi, R. S. H. Zade, M. J. Siavoshani, M. Saberian arXiv:1709.02656 [cs]. – arXiv, 2018. – URL: http://arxiv.org/abs/1709.02656 (date accessed: 04.01.2025). – Text : electronic.

26. Deep Learning for Encrypted Traffic Classification and Unknown Data Detection / M. H. Pathmaperuma, Y. Rahulamathavan, S. Dogan, et al. arXiv:2203.15501 [cs]. – arXiv, 2022. – URL: http://arxiv.org/abs/2203.15501 (date accessed: 04.01.2025). – Text : electronic.

27. Darknet Traffic Big-Data Analysis and Network Management for Real-Time Automating of the Malicious Intent Detection Process by a Weight Agnostic Neural Networks Framework / K. Demertzis, K. Tsiknas, D. Takezis et al. // Electronics. – 2021. – Vol. 10. – № 7. – P. 781.

28. Malicious Network Traffic Detection Based on Deep Neural Networks and Association Analysis / M. Gao, L. Ma, H. Liu et al. // Sensors. – 2020. – Vol. 20. – № 5. – P. 1452.

29. Improving neural networks by preventing co-adaptation of feature detectors / G. E. Hinton, N. Srivastava, A. Krizhevsky, et al. arXiv:1207.0580 [cs]. – arXiv, 2012. – URL: http://arxiv.org/abs/1207.0580 (date accessed: 04.01.2025). – Text : electronic.

30. Rumelhart, D. E. Learning Internal Representations by Error Propagation / D. E. Rumelhart, G. E. Hinton, R. J. Williams // MIT Press. – 1987. – P. 318-362.

31. Nair, V. Rectified linear units improve restricted boltzmann machines / V. Nair, G. E. Hinton. – Text : electronic // Proceedings of the 27th International Conference on International Conference on Machine Learning : ICML'10. – Madison, WI, USA : Omnipress, 2010. – P. 807-814. – URL: (date accessed: 04.01.2025).

32. Rumelhart, D. E. Learning representations by back-propagating errors / D. E. Rumelhart, G. E. Hinton, R. J. Williams // Nature. – 1986. – Vol. 323. – № 6088. – P. 533-536.

33. Glorot, X. Understanding the difficulty of training deep feedforward neural networks / X. Glorot, Y. Bengio. – Text : electronic // Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics Proceedings of the

Thirteenth International Conference on Artificial Intelligence and Statistics. – JMLR Workshop and Conference Proceedings, 2010. – Vol. 9. – P. 249-256. – URL: https://proceedings.mlr.press/v9/glorot10a.html (date accessed: 04.01.2025).

34. Prechelt, L. Early Stopping - But When? / L. Prechelt. – Text : electronic // Neural Networks: Tricks of the Trade : Lecture Notes in Computer Science / G. Goos et al. coll. ; G. B. Orr, K.-R. Müller eds. . – Berlin, Heidelberg : Springer Berlin Heidelberg, 1998. – Vol. 1524. – P. 55-69.

35. Dropout: A Simple Way to Prevent Neural Networks from Overfitting / N. Srivastava, G. Hinton, A. Krizhevsky et al. // Journal of Machine Learning Research. – 2014. – Vol. 15. – Dropout. – № 56. – P. 1929-1958.

36. Gradient-based learning applied to document recognition / Y. Lecun, L. Bottou, Y. Bengio, P. Haffner // Proceedings of the IEEE. – 1998. – Vol. 86. – № 11. – P. 2278-2324.

37. Hochreiter, S. Long Short-Term Memory / S. Hochreiter, J. Schmidhuber // Neural Comput. – 1997. – Vol. 9. – № 8. – P. 1735-1780.

38. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation / K. Cho, B. van Merrienboer, C. Gulcehre, et al. arXiv:1406.1078 [cs]. – arXiv, 2014. – URL: http://arxiv.org/abs/1406.1078 (date accessed: 04.01.2025). – Text : electronic.

39. Attention Is All You Need / A. Vaswani, N. Shazeer, N. Parmar, et al. arXiv:1706.03762 [cs]. – arXiv, 2023. – URL: http://arxiv.org/abs/1706.03762 (date accessed: 04.01.2025). – Text : electronic.

40. Definitions, methods, and applications in interpretable machine learning / W. J. Murdoch, C. Singh, K. Kumbier et al. // Proceedings of the National Academy of Sciences. – 2019. – Vol. 116. – № 44. – P. 22071-22080.

41. Kimura, M. New Perspective of Interpretability of Deep Neural Networks / M. Kimura, M. Tanaka. – Text : electronic // 2020 3rd International Conference on Information and Computer Technologies (ICICT) 2020 3rd International Conference on Information and Computer Technologies (ICICT). – San Jose, CA, USA : IEEE, 2020. –

P. 78-85. – URL: https://ieeexplore.ieee.org/document/9092045/ (date accessed: 04.01.2025).

42. Methods for interpreting and understanding deep neural networks // Digital Signal Processing. – 2018. – Vol. 73. – P. 1-15.

43. 6ns using Concept Induction / A. Dalal, M. K. Sarker, A. Barua, P. Hitzler arXiv:2301.09611 [cs]. – arXiv, 2023. – URL: http://arxiv.org/abs/2301.09611 (date accessed: 07.01.2025). – Text : electronic.

44. Improving Interpretability of Deep Neural Networks with Semantic Information / Y. Dong, H. Su, J. Zhu, B. Zhang arXiv:1703.04096 [cs]. – arXiv, 2017. – URL: http://arxiv.org/abs/1703.04096 (date accessed: 07.01.2025). – Text : electronic.

45. Interpretability of Deep Learning / Z. Huang, F. Li, Z. Wang, Z. Wang // International Journal of Future Computer and Communication. – 2022. – P. 34-39.

46. Interpretability vs. Complexity: The Friction in Deep Neural Networks / J. P. Amorim, P. H. Abreu, M. Reyes, J. Santos. – Text : electronic // 2020 International Joint Conference on Neural Networks (IJCNN) 2020 International Joint Conference on Neural Networks (IJCNN). – 2020. – Interpretability vs. Complexity. – P. 1-7. – URL: https://ieeexplore.ieee.org/abstract/document/9206800 (date accessed: 07.01.2025).

47. Layer-Wise Interpretation of Deep Neural Networks Using Identity Initialization / S. Kubota, H. Hayashi, T. Hayase, S. Uchida arXiv:2102.13333 [cs]. – arXiv, 2021. – URL: http://arxiv.org/abs/2102.13333 (date accessed: 07.01.2025). – Text : electronic.

48. Friedman, J. H. Greedy function approximation: A gradient boosting machine. / J. H. Friedman // The Annals of Statistics. – 2001. – Vol. 29. – Greedy function approximation. – № 5. – P. 1189-1232.

49. Ribeiro, M. T. “Why Should I Trust You?”: Explaining the Predictions of Any Classifier / M. T. Ribeiro, S. Singh, C. Guestrin. – Text : electronic // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining : KDD ’16. – New York, NY, USA : Association for Computing Machinery, 2016. – "Why Should I Trust You? – P. 1135-1144. – URL: https://dl.acm.org/doi/10.1145/2939672.2939778 (date accessed: 04.01.2025).

50. Lundberg, S. M. A Unified Approach to Interpreting Model Predictions / S. M. Lundberg, S.-I. Lee. – Text : electronic // Advances in Neural Information Processing Systems. – Curran Associates, Inc., 2017. – Vol. 30. – URL: https://papers.nips.cc/paper_files/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html (date accessed: 04.01.2025).

51. Ribeiro, M. T. Anchors: High-Precision Model-Agnostic Explanations / M. T. Ribeiro, S. Singh, C. Guestrin. – Text : electronic // Proceedings of the AAAI Conference on Artificial Intelligence. – 2018. – Vol. 32. – Anchors. – № 1. – URL: https://ojs.aaai.org/index.php/AAAI/article/view/11491 (date accessed: 04.01.2025).

52. Sundararajan, M. Axiomatic Attribution for Deep Networks / M. Sundararajan, A. Taly, Q. Yan. – Text : electronic // Proceedings of the 34th International Conference on Machine Learning International Conference on Machine Learning. – PMLR, 2017. – P. 3319-3328. – URL: https://proceedings.mlr.press/v70/sundararajan17a.html (date accessed: 04.01.2025).

53. Breiman L. Random Forests / Breiman L // Machine Learning. – 2001. – Vol. 45. – № 1. – P. 5-32.

54. Beyond Accuracy: The Role of Mental Models in Human-AI Team Performance / G. Bansal, B. Nushi, E. Kamar et al. // Proceedings of the AAAI Conference on Human Computation and Crowdsourcing. – 2019. – Vol. 7. – Beyond Accuracy. – P. 2-11.

55. User Trust Dynamics: An Investigation Driven by Differences in System Performance / K. Yu, S. Berkovsky, R. Taib et al. – Text : electronic // Proceedings of the 22nd International Conference on Intelligent User Interfaces : IUI ’17. – New York, NY, USA : Association for Computing Machinery, 2017. – User Trust Dynamics. – P. 307-317. – URL: https://doi.org/10.1145/3025171.3025219 (date accessed: 04.01.2025).

56. Human Evaluation of Models Built for Interpretability / I. Lage, E. Chen, J. He et al. // Proceedings of the AAAI Conference on Human Computation and Crowdsourcing. – 2019. – Vol. 7. – P. 59-67.

57. Lakkaraju, H. Interpretable Decision Sets: A Joint Framework for Description and Prediction / H. Lakkaraju, S. H. Bach, J. Leskovec. – Text : electronic // Proceedings

of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining : KDD ’16. – New York, NY, USA : Association for Computing Machinery, 2016. – Interpretable Decision Sets. – P. 1675-1684. – URL: https://doi.org/10.1145/2939672.2939874 (date accessed: 04.01.2025).

58. Making machine learning useable by revealing internal states update – a transparent approach / J. Zhou, M. A. Khawaja, Z. Li et al.

59. Reading Tea Leaves: How Humans Interpret Topic Models / J. Chang, S. Gerrish, C. Wang et al. – Text : electronic // Advances in Neural Information Processing Systems. – Curran Associates, Inc., 2009. – Vol. 22. – Reading Tea Leaves. – URL: https://papers.nips.cc/paper_files/paper/2009/hash/f92586a25bb3145facd64ab20fd554ff-Abstract.html (date accessed: 04.01.2025).

60. Towards better understanding of gradient-based attribution methods for Deep Neural Networks / M. Ancona, E. Ceolini, C. Öztireli, M. Gross arXiv:1711.06104 [cs]. – arXiv, 2018. – URL: http://arxiv.org/abs/1711.06104 (date accessed: 04.01.2025). – Text : electronic.

61. A Benchmark for Interpretability Methods in Deep Neural Networks / S. Hooker, D. Erhan, P.-J. Kindermans, B. Kim. – Text : electronic // Advances in Neural Information Processing Systems. – Curran Associates, Inc., 2019. – Vol. 32. – URL: https://papers.nips.cc/paper_files/paper/2019/hash/fe4b8556000d0f0cae99daa5c5c5a410-Abstract.html (date accessed: 04.01.2025).

62. Alvarez-Melis, D. Towards Robust Interpretability with Self-Explaining Neural Networks / D. Alvarez-Melis, T. S. Jaakkola arXiv:1806.07538 [cs]. – arXiv, 2018. – URL: http://arxiv.org/abs/1806.07538 (date accessed: 04.01.2025). – Text : electronic.

63. Molnar, C. Quantifying Model Complexity via Functional Decomposition for Better Post-hoc Interpretability / C. Molnar, G. Casalicchio, B. Bischl. – Text : electronic // Communications in Computer and Information Science / P. Cellier, K. Driessens eds. Book Title: Machine Learning and Knowledge Discovery in DatabasesDOI: 10.1007/978-3-030-43823-4_17. – Cham : Springer International Publishing, 2020. –

Vol. 1167. – P. 193-204. – URL: https://link.springer.com/10.1007/978-3-030-43823-4_17 (date accessed: 04.01.2025).

64. Nguyen, A. On quantitative aspects of model interpretability / A. Nguyen, M. R. Martínez arXiv:2007.07584 [cs]. – arXiv, 2020. – URL: http://arxiv.org/abs/2007.07584 (date accessed: 04.01.2025). – Text : electronic.

65. Robustness evaluation and robust design for proportional-integral-plus control / E. D. Wilson, Q. Clairon, R. Henderson, C. J. Taylor // International Journal of Control. – 2019. – Vol. 92. – № 12. – P. 2939-2951.

66. Alvarez Melis, D. Towards Robust Interpretability with Self-Explaining Neural Networks / D. Alvarez Melis, T. Jaakkola. – Text : electronic // Advances in Neural Information Processing Systems. – Curran Associates, Inc., 2018. – Vol. 31. – URL:https://proceedings.neurips.cc/paper_files/paper/2018/hash/3e9f0fc9b2f89e043bc6233994dfcf76-Abstract.html (date accessed: 04.01.2025).

67. Adversarial robustness in deep neural networks based on variable attributes of the stochastic ensemble model / R. Qin, L. Wang, X. Du et al. // Frontiers in Neurorobotics. – 2023. – Vol. 17.

68. Carlini, N. Towards Evaluating the Robustness of Neural Networks / N. Carlini, D. Wagner arXiv:1608.04644 [cs]. – arXiv, 2017. – URL: http://arxiv.org/abs/1608.04644 (date accessed: 04.01.2025). – Text : electronic.

69. Towards Deep Learning Models Resistant to Adversarial Attacks / A. Madry, A. Makelov, L. Schmidt, et al. arXiv:1706.06083 [stat]. – arXiv, 2019. – URL: http://arxiv.org/abs/1706.06083 (date accessed: 04.01.2025). – Text : electronic.

70. Intriguing properties of neural networks / C. Szegedy, W. Zaremba, I. Sutskever, et al. arXiv:1312.6199 [cs]. – arXiv, 2014. – URL: http://arxiv.org/abs/1312.6199 (date accessed: 04.01.2025). – Text : electronic.

71. ROBY: a Tool for Robustness Analysis of Neural Network Classifiers / P. Arcaini, A. Bombarda, S. Bonfanti, A. Gargantini. – Text : electronic // 2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST) 2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST). – Porto de

Galinhas, Brazil : IEEE, 2021. – ROBY. – P. 442-447. – URL: https://ieeexplore.ieee.org/document/9438585/ (date accessed: 04.01.2025).

72. Mitigating Adversarial Effects Through Randomization / C. Xie, J. Wang, Z. Zhang, et al. arXiv:1711.01991 [cs]. – arXiv, 2018. – URL: http://arxiv.org/abs/1711.01991 (date accessed: 04.01.2025). – Text : electronic.

73. Shorten, C. A survey on Image Data Augmentation for Deep Learning / C. Shorten, T. M. Khoshgoftaar // Journal of Big Data. – 2019. – Vol. 6. – № 1. – P. 60.

74. Jasser, J. Resilience from Diversity: Population-based approach to harden models against adversarial attacks. Resilience from Diversity / J. Jasser, I. Garibay arXiv:2111.10272 [cs]. – arXiv, 2022. – URL: http://arxiv.org/abs/2111.10272 (date accessed: 07.01.2025). – Text : electronic.

75. Lee, H. J. Advancing Adversarial Training by Injecting Booster Signal / H. J. Lee, Y. Yu, Y. M. Ro // IEEE Transactions on Neural Networks and Learning Systems. – 2024. – Т. 35. – № 9. – P. 12665-12677.

76. Hidden Technical Debt in Machine Learning Systems / D. Sculley, G. Holt, D. Golovin et al. – Text : electronic // Advances in Neural Information Processing Systems. – Curran Associates, Inc., 2015. – Vol. 28. – URL: https://proceedings.neurips.cc/paper_files/paper/2015/hash/86df7dcfd896fcaf2674f757a2463eba-Abstract.html (date accessed: 04.01.2025).

77. Data Validation for Machine Learning / E. Breck, N. Polyzotis, S. Roy et al.

78. Agrahari Supriya. Concept Drift Detection in Data Stream Mining : A literature review / Agrahari Supriya, Singh Anil. – Text : electronic // Journal of King Saud University - Computer and Information Sciences. – 2021. – Concept Drift Detection in Data Stream Mining. – № 34. – URL: https://www.researchgate.net/publication/356751987_Concept_Drift_Detection_in_Data_Stream_Mining_A_literature_review (date accessed: 04.01.2025).

79. Мазурок, Д. В. Мониторинг нейросетевых моделей: подходы и инструменты / Д. В. Мазурок // Наука и глобальные вызовы: перспективы развития : сборник статей IX Международной научно-практической конференции, Саратов,

20 июня 2024 года. – Саратов: Научно-образовательная платформа «Цифровая Наука», 2024. – С. 43-50.

80. ATMSeer: Increasing Transparency and Controllability in Automated Machine Learning / Q. Wang, Y. Ming, Z. Jin et al. // Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. – 2019. – ATMSeer. – P. 1-12.

81. Evidently AI - AI Observability and ML Monitoring Platform. – URL: https://www.evidentlyai.com/ (date accessed: 04.01.2025). – Text : electronic.

82. AI всё ещё легко обмануть - почему это нужно срочно менять. – URL: https://habr.com/ru/companies/ods/articles/871256/ (дата обращения: 12.10.2018). – Текст : электронный.

83. Adversarial Examples: Attacks and Defenses for Deep Learning / X. Yuan, P. He, Q. Zhu, X. Li // IEEE Transactions on Neural Networks and Learning Systems. – 2019. – Vol. 30. – Adversarial Examples. – № 9. – P. 2805-2824.

84. Хмелёва, А. А. Проблема компрометации системы распознавания изображений путем целенаправленной фальсификации обучающего множества / А. А. Хмелёва, Р. Ю. Демина, И. М. Ажмухамедов. – Текст : электронный // МОДЕЛИРОВАНИЕ, ОПТИМИЗАЦИЯ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ. – 2024. – Т. 12. – № 2(45). – URL: https://moitvivt.ru/ru/journal/pdf?id=1535 (дата обращения: 07.01.2025).

85. Adversarial Attack Type I: Cheat Classifiers by Significant Changes. Adversarial Attack Type I / S. Tang, X. Huang, M. Chen, et al. arXiv:1809.00594 [cs]. – arXiv, 2019. – URL: http://arxiv.org/abs/1809.00594 (date accessed: 07.01.2025). – Text : electronic.

86. Biggio, B. Poisoning Attacks against Support Vector Machines / B. Biggio, B. Nelson, P. Laskov arXiv:1206.6389 [cs]. – arXiv, 2013. – URL: http://arxiv.org/abs/1206.6389 (date accessed: 04.01.2025). – Text : electronic.

87. IWA: Integrated Gradient based White-box Attacks for Fooling Deep Neural Networks. IWA / Y. Wang, J. Liu, X. Chang, et al. arXiv:2102.02128 [cs]. – arXiv, 2021. – URL: http://arxiv.org/abs/2102.02128 (date accessed: 07.01.2025). – Text : electronic.

88. Carlini, N. MagNet and “Efficient Defenses Against Adversarial Attacks” are Not Robust to Adversarial Examples / N. Carlini, D. Wagner arXiv:1711.08478 [cs]. – arXiv, 2017. – URL: http://arxiv.org/abs/1711.08478 (date accessed: 04.01.2025). – Text : electronic.

89. Universal Adversarial Perturbations / S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard. – Text : electronic // 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – Honolulu, HI : IEEE, 2017. – P. 86-94. – URL: http://ieeexplore.ieee.org/document/8099500/ (date accessed: 04.01.2025).

90. ZOO: Zeroth Order Optimization based Black-box Attacks to Deep Neural Networks without Training Substitute Models / P.-Y. Chen, H. Zhang, Y. Sharma et al. – Text : electronic // Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security / arXiv:1708.03999 [stat]. – 2017. – ZOO. – P. 15-26. – URL: http://arxiv.org/abs/1708.03999 (date accessed: 04.01.2025).

91. Su, J. One pixel attack for fooling deep neural networks / J. Su, D. V. Vargas, S. Kouichi // IEEE Transactions on Evolutionary Computation. – 2019. – Vol. 23. – № 5. – P. 828-841.

92. Rozsa, A. Adversarial Diversity and Hard Positive Generation / A. Rozsa, E. M. Rudd, T. E. Boult. – Text : electronic // 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). – Las Vegas, NV, USA : IEEE, 2016. – P. 410-417. – URL: http://ieeexplore.ieee.org/document/7789548/ (date accessed: 04.01.2025).

93. Paleka, D. A law of adversarial risk, interpolation, and label noise / D. Paleka, A. Sanyal arXiv:2207.03933 [stat]. – arXiv, 2023. – URL: http://arxiv.org/abs/2207.03933 (date accessed: 04.01.2025). – Text : electronic.

94. Krizhevsky, A. Learning Multiple Layers of Features from Tiny Images / A. Krizhevsky.

95. ImageNet: A large-scale hierarchical image database / J. Deng, W. Dong, R. Socher et al. – Text : electronic // 2009 IEEE Conference on Computer Vision and Pattern

Recognition 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops). – Miami, FL : IEEE, 2009. – ImageNet. – P. 248-255. – URL: https://ieeexplore.ieee.org/document/5206848/ (date accessed: 04.01.2025).

96. Ptacek, T. H. Insertion Evasion and Denial of Service Eluding Network Intrusion Detection / T. H. Ptacek, T. N. Newsham.

97. Ibitoye, O. Analyzing Adversarial Attacks Against Deep Learning for Intrusion Detection in IoT Networks / O. Ibitoye, O. Shafiq, A. Matrawy arXiv:1905.05137 [cs]. – arXiv, 2019. – URL: http://arxiv.org/abs/1905.05137 (date accessed: 04.01.2025). – Text : electronic.

98. Adversarial Attacks on SDN-Based Deep Learning IDS System / Huang Chi-Hsuan, Lee Tsung-Han, Chang Lin-huang et al. // International Journal of Artificial Intelligence and Expert Systems (IJAE). – 2019. – Vol. 8. – Adversarial Attacks on SDN-Based Deep Learning IDS System. – № 3. – P. 181-191.

99. Rallying Adversarial Techniques against Deep Learning for Network Security / J. Clements, Y. Yang, A. Sharma, et al. arXiv:1903.11688 [cs]. – arXiv, 2021. – URL: http://arxiv.org/abs/1903.11688 (date accessed: 04.01.2025). – Text : electronic.

100. Wang, Z. Deep Learning-Based Intrusion Detection With Adversaries / Z. Wang // IEEE Access. – 2018. – Vol. 6. – P. 38367-38384.

101. Rigaki, M. Bringing a GAN to a Knife-Fight: Adapting Malware Communication to Avoid Detection / M. Rigaki, S. Garcia. – Text : electronic // 2018 IEEE Security and Privacy Workshops (SPW) 2018 IEEE Security and Privacy Workshops (SPW). – San Francisco, CA : IEEE, 2018. – Bringing a GAN to a Knife-Fight. – P. 70-75. – URL: https://ieeexplore.ieee.org/document/8424635/ (date accessed: 04.01.2025).

102. Sebastian Garcia. Virus Bulletin :: Modelling the network behaviour of malware to block malicious patterns. The Stratosphere Project: a behavioural IPS. – URL: https://www.virusbulletin.com/conference/vb2015/abstracts/modelling-network-behaviour-malware-block-malicious-patterns-stratosphere-project-behavioural-ips (date accessed: 04.01.2025). – Text : electronic.

103. Villamarin-Salomon, R. Identifying Botnets Using Anomaly Detection Techniques Applied to DNS Traffic / R. Villamarin-Salomon, J. C. Brustoloni. – Text : electronic // 2008 5th IEEE Consumer Communications and Networking Conference 2008 5th IEEE Consumer Communications and Networking Conference. – Las Vegas, Nevada, USA : IEEE, 2008. – P. 476-481. – URL: http://ieeexplore.ieee.org/document/4446410/ (date accessed: 04.01.2025).

104. Fomin, O. O. Assessment of the quality of neural network models based on a multifactorial information criterion / O. O. Fomin, V. A. Krykun // Herald of Advanced Information Technology. – 2024. – Vol. 7. – № 1. – P. 13-23.

105. What is Relevant Information? – SuperfastCPA CPA Review. What is Relevant Information? – URL: https://www.superfastcpa.com/what-is-relevant-information/ (date accessed: 04.01.2025). – Text : electronic.

106. Мазурок, Д. В. Повышение информативности нейросетевых анализаторов сетевого трафика при принятии решений / Д. В. Мазурок // Научный аспект. – 2024. – Т. 17, № 8. – С. 2084-2088.

107. Мазурок, Д. В. Подход к оценке робастности модели нейросетевого классификатора / Д. В. Мазурок, А. В. Рунов // Наука, технологии и инновации: стратегии развития в современном мире : сборник статей II Международной научно-практической конференции, Москва, 23 апреля 2024 года. – Москва: Издательство "Доброе слово и Ко", 2024. – С. 137-146.

108. Improving Security of Neural Networks in the Identification Module of Decision Support Systems / Y. Monakhov, M. Monakhov, A. Telny et al. – Text : electronic // 2020 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT) 2020 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT). – Yekaterinburg, Russia : IEEE, 2020. – P. 571-574

109. Мазурок, Д. В. Мониторинг нейросетевых моделей: подходы и инструменты / Д. В. Мазурок // Исследования и инновации: синергия знаний и практики : сборник статей II Международной научно-практической конференции,

Самара, 20 февраля 2025 года. – Москва: Издательство "Доброе слово и Ко", 2025. – С. 102-109.

110. Мазурок, Д. В. Составляющие оценки качества модели нейросетевого классификатора / Д. В. Мазурок // Наука и глобальные вызовы: перспективы развития : сборник статей VIII Международной научно-практической конференции, Саратов, 30 мая 2024 года. – Саратов: Научно-образовательная платформа «Цифровая Наука», 2024. – С. 62-67.

111. Towards Detecting and Classifying Network Intrusion Traffic Using Deep Learning Frameworks / R. B. Basnet, R. Shash, C. Johnson et al. // Journal of Internet Services and Information Security. – 2019. – Vol. 9. – № 4. – P. 1-17.

112. Мазурок, Д. В. Мультиагентный подход анализа результатов интерпретации с применением больших языковых моделей / Д. В. Мазурок, М. Ю. Монахов // Наука, технологии и инновации: стратегии развития в современном мире : сборник статей II Международной научно-практической конференции, Москва, 23 апреля 2024 года. – Москва: Издательство "Доброе слово и Ко", 2024. – С. 130-136.

113. AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty. AugMix / D. Hendrycks, N. Mu, E. D. Cubuk, et al. arXiv:1912.02781 [stat]. – arXiv, 2020. – URL: http://arxiv.org/abs/1912.02781 (date accessed: 04.01.2025). – Text : electronic.

114. Going Deeper with Embedded FPGA Platform for Convolutional Neural Network / J. Qiu, J. Wang, S. Yao et al. – Text : electronic // Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays : FPGA ’16. – New York, NY, USA : Association for Computing Machinery, 2016. – P. 26-35. – URL: https://doi.org/10.1145/2847263.2847265 (date accessed: 04.01.2025).

115. Randaugment: Practical automated data augmentation with a reduced search space / E. D. Cubuk, B. Zoph, J. Shlens, Q. V. Le. – Text : electronic // 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops

(CVPRW). – Seattle, WA, USA : IEEE, 2020. – Randaugment. – P. 3008-3017. – URL: https://ieeexplore.ieee.org/document/9150790/ (date accessed: 04.01.2025).

116. Learning both Weights and Connections for Efficient Neural Network / S. Han, J. Pool, J. Tran, W. Dally. – Text : electronic // Advances in Neural Information Processing Systems. – Curran Associates, Inc., 2015. – Vol. 28. – URL: https://papers.nips.cc/paper_files/paper/2015/hash/ae0eb3eed39d2bcef4622b2499a05fe6-Abstract.html (date accessed: 04.01.2025).

117. Ng Andrew Y. Feature selection, L1 vs. L2 regularization, and rotational invariance / Ng Andrew Y. – Text : electronic // Twenty-first international conference on Machine learning - ICML '04 Twenty-first international conference. – Banff, Alberta, Canada : ACM Press, 2004. – P. 78. – URL: http://portal.acm.org/citation.cfm?doid=1015330.1015435 (date accessed: 04.01.2025).

118. Focal Loss for Dense Object Detection / T.-Y. Lin, P. Goyal, R. Girshick, et al. arXiv:1708.02002 [cs]. – arXiv, 2018. – URL: http://arxiv.org/abs/1708.02002 (date accessed: 04.01.2025). – Text : electronic.

119. Haibo He. Learning from Imbalanced Data / Haibo He, E. A. Garcia // IEEE Transactions on Knowledge and Data Engineering. – 2009. – Vol. 21. – № 9. – P. 1263-1284.

120. On Calibration of Modern Neural Networks / C. Guo, G. Pleiss, Y. Sun, K. Q. Weinberger. – Text : electronic // Proceedings of the 34th International Conference on Machine Learning International Conference on Machine Learning. – PMLR, 2017. – P. 1321-1330. – URL: https://proceedings.mlr.press/v70/guo17a.html (date accessed: 04.01.2025).

121. Монахов, Ю. М. Функциональная устойчивость информационных систем : учебное пособие : ч. 3 / Монахов, Ю. М., Монахов М. Ю. – Издательство ВлГУ. – 2011. – Вып. 1. – 60 с.

122. Goodfellow, I. J. Explaining and Harnessing Adversarial Examples / I. J. Goodfellow, J. Shlens, C. Szegedy arXiv:1412.6572 [stat]. – arXiv, 2015. – URL: http://arxiv.org/abs/1412.6572 (date accessed: 04.01.2025). – Text : electronic.

123. Moosavi-Dezfooli, S.-M. DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks / S.-M. Moosavi-Dezfooli, A. Fawzi, P. Frossard. – Text : electronic // 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – Las Vegas, NV, USA : IEEE, 2016. – DeepFool. – P. 2574-2582. – URL: http://ieeexplore.ieee.org/document/7780651/ (date accessed: 04.01.2025).

ПРИЛОЖЕНИЕ 1

Перечень отобранных признаков для экспериментов

dst_port – порт назначения,

protocol – используемый протокол,

flow_duration – длительность соединения,

tot_fwd_pkts – общее количество пакетов, отправленных от источника,

tot_bwd_pkts – общее количество пакетов, полученных от источника,

totlen_fwd_pkts – общий объем данных (в байтах), отправленных от источника,

totlen_bwd_pkts – общий объем данных (в байтах), полученных от источника,

fwd_pkt_len_mean – средняя длина пакета, отправленного от источника,

fwd_pkt_len_std – стандартное отклонение длины пакетов, отправленных от источника,

bwd_pkt_len_mean – средняя длина пакета, полученного от источника,

flow_byts_s – количество байтов на поток в секунду,

flow_pkts_s – количество пакетов на поток в секунду,

flow_iat_std – стандартное отклонение межпакетного интервала в потоке,

flow_iat_min – минимальный межпакетный интервал в потоке,

fwd_iat_tot – общий межпакетный интервал для исходящего трафика,

fwd_iat_min – минимальный межпакетный интервал для исходящего трафика,

bwd_iat_tot – общий межпакетный интервал для входящего трафика,

bwd_iat_min – минимальный межпакетный интервал для входящего трафика,

fwd_psh_flags – количество флагов PUSH в исходящем трафике,

fwd_urg_flags – количество флагов URG в исходящем трафике,

bwd_pkts_s – количество пакетов на входящий поток в секунду,

fin_flag_cnt – наличие флагов FIN в потоке,

rst_flag_cnt – наличие флагов RST в потоке,

psh_flag_cnt – наличие флагов PSH в потоке,

ack_flag_cnt – наличие флагов ACK в потоке,

urg_flag_cnt – наличие флагов URG в потоке,

down_up_ratio – отношение входящего и исходящего трафика,

init_fwd_win_byts – размер начального окна для исходящего трафика в байтах,

init_bwd_win_byts – размер начального окна для входящего трафика в байтах,

fwd_seg_size_min – минимальный размер сегмента для исходящего трафика,

active_mean – средняя продолжительность активных периодов соединения,

idle_mean – средняя продолжительность неактивных периодов соединения.

ПРИЛОЖЕНИЕ 2

Используемые промпты для мультиагентного взаимодействия

DLExpert:

You are the DLExpert. Your task is to analyze the Integrated Gradient values to determine if the neural network's classification seems reasonable.

Assess whether the Integrated Gradient interpretation scores align with the expected feature importance and neural network behavior.

Also analyze the features provided as inputs to the neural network.

Understand the significance of each feature, their typical values, and identify any unusual feature combinations.

Remember - the feature set comes from IDS2018 dataset.

Identify any discrepancies or potential issues with the neural network's decision-making process.

Remember that trained model may make mistakes and could potentially trigger on noise or unusual feature combinations.

Provide a thorough yet concise report with your findings. Summarizing your findings in a clear and actionable manner, focusing on the most critical issues observed.

Provide no recommendations or any other extra info - remember, the ultimate goal is to identify if the classification result is FP or TP! {task}

NetSecExpert:

You are the NetSecExpert. Your task is to analyze the feature values to determine if these feature combinations seem casual or attack-like.

You are highly experienced with Net Security and understand all patterns and details of any type of network attack that exists in the IDS2018 dataset.

Provide a thorough yet concise report with your findings. Summarizing your findings in a clear and actionable manner, focusing on the most critical issues observed."

Provide no recommendations or any other extra info - remember, the ultimate goal is to identify if the classification result is FP or TP! DATA: {task['input_values']}

Judger:

Based on the following reports from DLExpert, and NetSecExpert, make a final decision on whether the neural network's classification is a true positive (TP) or false positive (FP).

Provide a well-argued and supported decision, please be concise!"

REPORTS: {name}: {report}

ПРИЛОЖЕНИЕ 3

Полная таблица результатов этапа 2 эксперимента

Участник	TP	TN	FP	FN	P	R	F1
1	26	21	9	4	0,743	0,867	0,800
2	26	23	7	4	0,788	0,867	0,825
3	26	23	7	4	0,788	0,867	0,825
4	25	24	6	5	0,806	0,833	0,820
5	26	23	7	4	0,788	0,867	0,825
6	26	23	7	4	0,788	0,867	0,825
7	26	23	7	4	0,788	0,867	0,825
8	26	23	7	4	0,788	0,867	0,825
9	25	22	8	5	0,758	0,833	0,794
10	26	22	8	4	0,765	0,867	0,812
11	24	23	7	6	0,774	0,800	0,787
12	26	23	7	4	0,788	0,867	0,825
13	26	23	7	4	0,788	0,867	0,825
14	24	23	7	6	0,774	0,800	0,787
15	23	22	8	7	0,742	0,767	0,754
16	26	23	7	4	0,788	0,867	0,825
17	26	23	7	4	0,788	0,867	0,825
18	25	23	7	5	0,781	0,833	0,806
19	26	23	7	4	0,788	0,867	0,825
20	25	24	6	5	0,806	0,833	0,820
21	27	21	9	3	0,750	0,900	0,818
22	26	23	7	4	0,788	0,867	0,825
23	27	21	9	3	0,750	0,900	0,818
24	23	23	7	7	0,767	0,767	0,767
25	25	23	7	5	0,781	0,833	0,806
26	25	22	8	5	0,758	0,833	0,794
27	26	23	7	4	0,788	0,867	0,825
28	25	24	6	5	0,806	0,833	0,820
29	25	24	6	5	0,806	0,833	0,820
30	25	22	8	5	0,758	0,833	0,794
31	24	24	6	6	0,800	0,800	0,800
32	23	23	7	7	0,767	0,767	0,767
33	26	23	7	4	0,788	0,867	0,825
34	24	21	9	6	0,727	0,800	0,762
35	25	22	8	5	0,758	0,833	0,794
36	24	24	6	6	0,800	0,800	0,800
37	25	22	8	5	0,758	0,833	0,794
38	23	22	8	7	0,742	0,767	0,754

Полная таблица результатов этапа 2 эксперимента. Продолжение

39	23	22	8	7	0,742	0,767	0,754
40	25	21	9	5	0,735	0,833	0,781
41	25	24	6	5	0,806	0,833	0,820
42	24	25	5	6	0,828	0,800	0,814
43	25	20	10	5	0,714	0,833	0,769
44	25	24	6	5	0,806	0,833	0,820
45	26	23	7	4	0,788	0,867	0,825
46	24	24	6	6	0,800	0,800	0,800
47	26	23	7	4	0,788	0,867	0,825
48	19	21	9	11	0,679	0,633	0,655
49	25	23	7	5	0,781	0,833	0,806
50	24	24	6	6	0,800	0,800	0,800
51	25	23	7	5	0,781	0,833	0,806
52	25	23	7	5	0,781	0,833	0,806
53	19	22	8	11	0,704	0,633	0,667
54	26	23	7	4	0,788	0,867	0,825
55	25	24	6	5	0,806	0,833	0,820
56	26	23	7	4	0,788	0,867	0,825
57	26	23	7	4	0,788	0,867	0,825
58	25	23	7	5	0,781	0,833	0,806
59	26	23	7	4	0,788	0,867	0,825
60	23	22	8	7	0,742	0,767	0,754
Ст. откл	1,504	0,976	0,976	1,504	0,028	0,050	0,034
Среднее	24,900	22,783	7,217	5,100	0,775	0,830	0,801
Медиана	25,000	23,000	7,000	5,000	0,788	0,833	0,813
LLM	25	23	7	5	0,781	0,833	0,806

ПРИЛОЖЕНИЕ 4

Код расчета оценки адаптируемости

```
def calculate_adaptability(df1: pd.DataFrame, df2: pd.DataFrame, monitoring_enabled: bool = False, e: float = 1e-2):
    """
    Функция для расчета адаптируемости модели на основе сравнения распределений признаков в двух наборах с использованием статистики Колмогорова-Смирнова.
    :param df1: обучающий набор
    :param df2: тестовый набор (промышленный набор)
    :param e: Константа для сглаживания (по умолчанию 1e-2)
    :return: Значение адаптируемости модели
    """
    # Проверка: у двух DataFrame должны совпадать названия колонок
    if list(df1.columns) != list(df2.columns):
        raise ValueError("Оба DataFrame должны иметь одинаковые колонки (признаки)")
    p_values = []
    # Цикл по каждому признаку для сравнения распределений
    for column in df1.columns:
        data1 = df1[column]
        data2 = df2[column]
        # Проверяем, является ли признак категориальным или числовым
        if data1.nunique() > 2:  Если числовой, применяем тест Колмогорова-Смирнова
            stat, p_value = ks_2samp(data1, data2)
            print(f"Признак: {column} (числовой) Статистика К-С: {stat} p-value: {p_value}")
        else: # Если категориальный, применяем тест Хи-квадрат
            contingency_table = pd.crosstab(data1, data2)
            chi2, p_value, _, _ = chi2_contingency(contingency_table)
            print(f"Признак: {column} Статистика Хи-квадрат: {chi2} p-value: {p_value}")
        p_values.append(p_value)
    p_sup = np.max(1 - np.array(p_values))
    print(f"Супремум p-value: { p_sup}")
    adaptability = max(int(monitoring_enabled), (-np.log(p_sup + e)) / (-np.log(e)))
    return adaptability
```

ПРИЛОЖЕНИЕ 5

Код расчета падения уверенности на FGSM

```
def calculate_certainty_drop_batches(model, test_loader, epsilon=0.05, device='cpu'):
    """
    Создает FGSM примеры и расчитывает падение уверенности в предсказании
    Args:
    - model (torch.nn.Module): The trained model.
    - test_loader (torch.utils.data.DataLoader): DataLoader containing the test dataset.
    - epsilon (float): The perturbation magnitude for FGSM.
    """
    model.eval()
    model.to(device)
    certainty_drops = []
    for data in test_loader:
        inputs, labels = data['inputs'].to(device), data['target'].to(device)
        inputs.requires_grad = True
        original_outputs = model(inputs)
        original_confidences = torch.softmax(original_outputs, dim=1)
        original_certainties = original_confidences.gather(1, labels.unsqueeze(1))
        criterion = nn.CrossEntropyLoss()
        loss = criterion(original_outputs, labels)
        model.zero_grad()
        loss.backward()
        perturbations = epsilon * inputs.grad.sign()
        adversarial_inputs = inputs + perturbations
        adv_samples.append(adversarial_inputs.detach().cpu())
        adversarial_outputs = model(adversarial_inputs)
        adversarial_confidences = torch.softmax(adversarial_outputs, dim=1)
        adversarial_certainties = adversarial_confidences.gather(1, labels.unsqueeze(1))
        batch_certainty_drops = original_certainties - adversarial_certainties
        certainty_drops.append(batch_certainty_drops)
    certainty_drops = torch.cat(certainty_drops, dim=0)
    return certainty_drops
```

ПРИЛОЖЕНИЕ 6

Основной код генерации аугментированных данных при помощи cVAE

```
# константы
FEATS = ['flow_duration', 'tot_fwd_pkts', 'tot_bwd_pkts',
        'totlen_fwd_pkts', 'totlen_bwd_pkts', 'fwd_pkt_len_mean',
        'fwd_pkt_len_std', 'bwd_pkt_len_mean', 'flow_byts_s', 'flow_pkts_s',
        'flow_iat_std', 'flow_iat_min', 'fwd_iat_tot', 'fwd_iat_min',
        'bwd_iat_tot', 'bwd_iat_min', 'fwd_psh_flags', 'fwd_urg_flags',
        'bwd_pkts_s', 'fin_flag_cnt', 'rst_flag_cnt', 'psh_flag_cnt',
        'ack_flag_cnt', 'urg_flag_cnt', 'down_up_ratio', 'init_fwd_win_byts',
        'init_bwd_win_byts', 'fwd_seg_size_min', 'active_mean', 'idle_mean',
        'dst_port_DHCP', 'dst_port_DNS', 'dst_port_FTP_and_Telnet',
        'dst_port_LDAP', 'dst_port_Mail', 'dst_port_NTP', 'dst_port_Other',
        'dst_port_Remote_Desktop', 'dst_port_SNMP', 'dst_port_Secure_Web',
        'dst_port_Web', 'dst_port_Database', 'protocol_HOPOPT', 'protocol_TCP',
        'protocol_UDP']
  FEATS_independent = ['flow_duration', 'tot_fwd_pkts', 'tot_bwd_pkts',
        'totlen_fwd_pkts', 'totlen_bwd_pkts',
        'fwd_pkt_len_std',
        'flow_iat_std', 'flow_iat_min', 'fwd_iat_tot', 'fwd_iat_min',
        'bwd_iat_tot', 'bwd_iat_min', 'fwd_psh_flags', 'fwd_urg_flags',
        'fin_flag_cnt', 'rst_flag_cnt', 'psh_flag_cnt',
        'ack_flag_cnt', 'urg_flag_cnt', 'init_fwd_win_byts',
        'init_bwd_win_byts', 'fwd_seg_size_min', 'active_mean', 'idle_mean',
        'dst_port_DHCP', 'dst_port_DNS', 'dst_port_FTP_and_Telnet',
        'dst_port_LDAP', 'dst_port_Mail', 'dst_port_NTP', 'dst_port_Other',
        'dst_port_Remote_Desktop', 'dst_port_SNMP', 'dst_port_Secure_Web',
        'dst_port_Web', 'dst_port_Database', 'protocol_HOPOPT', 'protocol_TCP',
        'protocol_UDP']
  stats = \
     {'flow_duration': {'mean': 12190203.434576446, 'std': 30887842.254683726},
      'tot_fwd_pkts': {'mean': 23.5906895037668, 'std': 1522.7240255516112},
      'tot_bwd_pkts': {'mean': 6.310124689346987, 'std': 163.3532601518622},
      'totlen_fwd_pkts': {'mean': 978.4489922267636, 'std': 64978.58289088951},
      'totlen_bwd_pkts': {'mean': 4725.147607788915, 'std': 234013.85726845363},
      'fwd_pkt_len_mean': {'mean': 50.312145, 'std': 60.379177},
      'fwd_pkt_len_std': {'mean': 70.8479, 'std': 116.40222},
      'bwd_pkt_len_mean': {'mean': 113.07786, 'std': 163.94756},
      'flow_byts_s': {'mean': 257061.99670678814, 'std': 3640032.4338215864},
      'flow_pkts_s': {'mean': 52292.43078478612, 'std': 263944.9741311093},
      'flow_iat_std': {'mean': 1212559.4, 'std': 289196500.0},
      'flow_iat_min': {'mean': 2980272.9026470724, 'std': 13773998.17681317},
      'fwd_iat_tot': {'mean': 11899648.614643246, 'std': 30808336.113964636},
      'fwd_iat_min': {'mean': 3068315.3384784604, 'std': 13933770.47828859},
      'bwd_iat_tot': {'mean': 7595239.169691662, 'std': 25859396.3880283},
      'bwd_iat_min': {'mean': 291710.7985332142, 'std': 3832044.5472794054},
      'fwd_psh_flags': {'mean': 0.04389661640210948, 'std': 0.20486509390910615},
```

```
            'fwd_urg_flags': {'mean': 0.0001646343538763844, 'std': 0.012829936168247662},
            'bwd_pkts_s': {'mean': 15312.658, 'std': 92415.88},
            'fin_flag_cnt': {'mean': 0.004801501637796105, 'std': 0.06912631617392533},
            'rst_flag_cnt': {'mean': 0.18740286919638877, 'std': 0.390234603206505},
            'psh_flag_cnt': {'mean': 0.3921353137300893, 'std': 0.48822661522181227},
            'ack_flag_cnt': {'mean': 0.33162410327024816, 'std': 0.47079674432639723},
            'urg_flag_cnt': {'mean': 0.0417306504966675, 'std': 0.1999730141422033},
            'down_up_ratio': {'mean': 0.49633422899144747, 'std': 1.0015967485593569},
            'init_fwd_win_byts': {'mean': 12089.785303533226, 'std': 15318.221866625474},
            'init_bwd_win_byts': {'mean': 17683.967169390915, 'std': 18630.214259308126},
            'fwd_seg_size_min': {'mean': 17.993538140112307, 'std': 7.693290998313641},
            'active_mean': {'mean': 172323.53, 'std': 2500096.2},
            'idle_mean': {'mean': 4963745.5, 'std': 220834530.0}}

    # нормализация данных
        def normalize_df(df, stats):
            normalized_df = df.copy().astype(dtypes, errors='ignore')
            for col, values in stats.items():
                mean = values['mean']
                std = values['std']
                normalized_df[col] = (normalized_df[col] - mean) / std
            return normalized_df

    # денормализация данных
        def denormalize_df(df, stats): # old!
            denormalized_df = df.copy()
            for col, values in stats.items():
                mean = values['mean']
                std = values['std']
                denormalized_df[col] = denormalized_df[col] * std + mean
            return denormalized_df.round(8).astype(dtypes, errors='ignore')

    # повторная параметризация
        def new_reparameterize(self, mu, logvar, perturbation_scale=.01):
            if True: #self.training:
                std = logvar.mul(0.5).exp_()
                eps = Variable(std.data.new(std.size()).normal_())
                return eps.mul(std).mul_(perturbation_scale).add_(mu)
            else:
                return mu

    # расчет зависимых признаков
        def calculate_dependent_features(flow_duration, tot_fwd_pkts, tot_bwd_pkts,
totlen_fwd_pkts,
                                         totlen_bwd_pkts, zero_flow_medians=zero_flow_medians):
            fl_ms = (flow_duration / 1000000)
            flow_byts_s = (totlen_fwd_pkts + totlen_bwd_pkts) / fl_ms if fl_ms else \
                                zero_flow_medians['flow_byts_s']
            flow_pkts_s = (tot_fwd_pkts + tot_bwd_pkts) / fl_ms if fl_ms else \
                                zero_flow_medians['flow_pkts_s']

            fwd_pkt_len_mean = totlen_fwd_pkts / tot_fwd_pkts if tot_fwd_pkts > 0 else 0
```

```
            bwd_pkt_len_mean = totlen_bwd_pkts / tot_bwd_pkts if tot_bwd_pkts > 0 else 0

            down_up_ratio = round(tot_bwd_pkts / tot_fwd_pkts) if tot_fwd_pkts > 0 else 0
            bwd_pkts_s = tot_bwd_pkts / (flow_duration / 1000)

            to_return = {
               'flow_byts_s': flow_byts_s,
               'flow_pkts_s': flow_pkts_s,
               'fwd_pkt_len_mean': fwd_pkt_len_mean,
               'bwd_pkt_len_mean': bwd_pkt_len_mean,
               'down_up_ratio': down_up_ratio,
               'bwd_pkts_s': bwd_pkts_s,
            }
            return to_return

         # коррекция значений зависимых признаков
            def correct_dependent(adversarial_sample, return_dict = False):
               # print(adversarial_sample)
               adversarial_sample_denormalized = denormalize_df(
                  pd.DataFrame(
                     {f:[i] for f, i in zip(FEATS, adversarial_sample.squeeze().detach().numpy())}
                  ), stats).to_dict()
               dependent_adversarial                                                                    =
calculate_dependent_features(adversarial_sample_denormalized['flow_duration'][],
                                    adversarial_sample_denormalized['tot_fwd_pkts'][],
                                    adversarial_sample_denormalized['tot_bwd_pkts'][],
                                    adversarial_sample_denormalized['totlen_fwd_pkts'][],
                                    adversarial_sample_denormalized['totlen_bwd_pkts'][])
               adversarial_sample_corrected = adversarial_sample_denormalized.copy()
               for k, v in dependent_adversarial.items():
                  adversarial_sample_corrected[k]=[v]
               if return_dict:
                  return {k:[v[]] for k, v in adversarial_sample_corrected.items()}
               res_ = {k:[v[]] for k, v in adversarial_sample_corrected.items()}
               norm_df = normalize_df(
                     pd.DataFrame(res_), stats)[selected_features_all_ordered]
               res = torch.from_numpy(norm_df.values).float()#.unsqueeze(0)
               return res
         # перевод данных в признаки, готовые для классификации
            def convert_to_FEATS(x_independent, FEATS, FEATS_independent):
               # print(FEATS)
               # create a numpy array with order 'FEATS' and fill missing values with -1
               x = np.full(len(FEATS), -1.0)
               for i, feat in enumerate(FEATS_independent):
                  j = FEATS.index(feat)
                  if i < len(x_independent) and not np.isnan(x_independent[i]):
                     x[j] = x_independent[i]
               return x

         # получение аугментированного примера данных при помощи обученной cVAE модели
            def generate_noisy_sample(cvae, original_data, condition, perturbation_scale=1.0):
               # Pass original data through the encoder to get latent space representation
```

```
            with torch.no_grad():
                mu, logvar = cvae.encode(original_data, condition)

                # Reparameterize
                std = torch.exp(0.5 * logvar)
                eps = torch.randn_like(std)
                z = mu + eps * std * perturbation_scale
                noisy_sample = cvae.decode(z, condition)
            results = []
            for augmented_sample in noisy_sample:
                x_hat = torch.cat((augmented_sample, condition[]), 0)

                result = correct_dependent(
                    torch.from_numpy(
                        convert_to_FEATS(x_hat.squeeze(0).numpy(), FEATS, FEATS_independent)
                    ),
                    False
                )
                results.append(result)
            return results
        # получение нужного cVAE модели представления тензоров
        def get_cvae_view(X, y):
            row = X
            target = y
            cvae_view = pd.DataFrame([np.concatenate((row, np.array([target])))], columns=selected_features_all_ordered+TARGET)[FEATS_independent+TARGET]
            return torch.from_numpy(cvae_view.values[0,:24]).float(), torch.from_numpy(cvae_view.values[0,24:]).long()

        # получение аугментированных данных при помощи обученной cVAE модели
        def generate_augmented_samples(test_loader, cvae_model, start_idx, end_idx, perturbation_scale=1.0):
            augmented_samples = []
            augmented_labels = []
            for i in tqdm.tqdm(range(start_idx, end_idx)):
                X, C = get_cvae_view(test_loader.dataset[i]['inputs'], test_loader.dataset[i]['target'])
                # используем CVAE
                noisy_sample = generate_noisy_sample(cvae_model, X.unsqueeze(0), C.unsqueeze(0), perturbation_scale=perturbation_scale)
                augmented_samples.append(noisy_sample)
                augmented_labels.append(test_loader.dataset[i]['target'].item())
            return augmented_samples, augmented_labels

        cvae_model.dropout = torch.nn.Dropout(p=0)
        cvae_model.reparameterize = types.MethodType(new_reparameterize, cvae_model)
        augmented_samples, augmented_labels = generate_augmented_samples(test_loader, cvae_model, start_idx, end_idx, perturbation_scale=1.0)
```

ПРИЛОЖЕНИЕ 7

Код DeepFool метода из эксперимента

```
def deepfool (model, inputs, labels, max_steps=50, class_weight=None, device='cpu'):
    """
    Создает состязательные примеры с использованием DeepFool.
    - model (torch.nn.Module): Обученная модель
    - inputs (torch.Tensor): Входной тензор
    - labels (torch.Tensor): GT метки
    - max_steps (int): Максимальное число итераций
    - class_weight (torch.Tensor, optional): Веса классов.
    Returns:
    - perturbed_inputs (torch.Tensor): Состязательные примеры
    - steps_to_cross (torch.Tensor): Число шагов (итераций)
    """
    model.eval()
    inputs = inputs.to(device)
    labels = labels.to(device)
    perturbed_inputs = inputs.clone().detach().requires_grad_(True)
    steps_to_cross = torch.zeros(inputs.size(0), device=device)
    for step in range(max_steps):
        outputs = model(perturbed_inputs)
        _, pred = outputs.max(1)
        not_crossed = pred.eq(labels).float()
        steps_to_cross += not_crossed
        if not_crossed.sum().item() == 0:
            break
        criterion = nn.CrossEntropyLoss(weight=class_weight)
        loss = criterion(outputs, labels)
        model.zero_grad()
        loss.backward()
        grad_sign = perturbed_inputs.grad.data.sign()
        perturbed_inputs = perturbed_inputs + 0.02 * grad_sign
        perturbed_inputs = torch.clamp(perturbed_inputs, 0, 1).detach().requires_grad_(True)
    return perturbed_inputs, steps_to_cross
```

ПРИЛОЖЕНИЕ 8

Акты о внедрении результатов диссертационного исследования

Общество с ограниченной ответственностью

СИСТЕМ ИНВЕСТ

ООО «НТЦ «Системинвест»

«Научно-технический центр Системинвест»

«STC SYSTEMINVEST»

119571, Москва, проспект Вернадского, 94, корп. 4, пом. II, ком. 5

www.systeminvest.ru

5-II-4-94, Vernadskogo pr., Moscow, 119571

info@systeminvest.ru

ИНН / КПП 7729486448 / 772901001

УТВЕРЖДАЮ
Генеральный директор
ООО «НТЦ «Системинвест»
Кожеватов Р.Д.
05.06.2024

АКТ

об использовании результатов
кандидатской диссертационной работы
Мазурка Дмитрия Валерьевича

Результаты диссертационной работы Мазурка Д.В. «Алгоритмы глубокого машинного обучения в системах анализа сетевого трафика» в виде предложенного подхода к повышению точности работы системы обнаружения сетевых вторжений с использованием Больших языковых моделей применяются при эксплуатации нейросетевых классификаторов в «НТЦ «Системинвест». Применение данного подхода в среднем повысило общую точность системы на 12% за счет автоматизированной проверки решений классификатора и их корректировки с использованием Большой языковой модели GPT 4.

Генеральный директор _____ Кожеватов Р.Д.

ООО «Войс Коммьюникэйшн»
Россия, Москва, 115114,
Переулок 1-й Кожевнический,
д.10, эт.пом. оф.цоколь I.4/1
+7 (495) 011-04-50
http://lab.voice-com.ru

VOICECOM

УТВЕРЖДАЮ
Директор ООО «Войс Коммьюникэйшн»
/ Полковникова С.В.
«05» июня 2024 г.

АКТ

о практическом применении результатов диссертационного исследования Мазурка Дмитрия Валерьевича «Алгоритмы глубокого машинного обучения в системах анализа сетевого трафика»

Модуль интерпретации решений нейросетевого классификатора был внедрен и практически используется прогнозировании сетевой нагрузки центра обработки данных **ВойсКоммьюникэйшн».**

Применение данного модуля позволило провести более глубокий анализ используемых подходов прогнозирования, что повысило эффективность работы сети в условиях пиковой нагрузки в среднем на 15% за счет адекватного предиктивного масштабирования, перераспределения ресурсов.

IT Директор
ООО «Войс Коммьюникэйшн» Денисов Ю.Б.

ЮКИТЕХ ЛАБ

УТВЕРЖДАЮ

Генеральный директор

«ООО ЮКИТЕХ ЛАБ»

_____Красюк В.Ю.

03 июня 2024 года

АКТ

об использовании результатов

кандидатской диссертационной работы

Мазурка Дмитрия Валерьевича

Результаты диссертационной работы Мазурка Д.В. «Алгоритмы глубокого машинного обучения в системах анализа сетевого трафика» в виде предложенного подхода к повышению точности работы системы обнаружения сетевых вторжений с использованием Больших языковых моделей применятся при эксплуатации нейросетевых классификаторов в ООО «ЮКИТЕХ ЛАБ». Применение данного подхода в среднем повысило общую точность системы на 12% за счет автоматизированной проверки решений классификатора и их корректировки с использованием Большой языковой модели GPT 4.

Технический директор ООО «ЮКИТЕХ ЛАБ» _____ Юркин Г.И.

03 июня 2024 года

LLC ItArt house 12 Lesnaya st. Govorovo village
inside city area.Moskovskiy. Moscow city 119618
www.itart.me
+7 (499) 390 55 85

Общество с ограниченной ответственностью «АйТиАрт»
119618 г.Москва, вн.тер.г.поселение Московский, д.Говорово, ул.Лесная, д.12 ИНН/КПП 7729704784/775101001

УТВЕРЖДАЮ
Генеральный директор
ООО «АйТиАрт»
_____ / Варёнова С.В.
« 02 » июня 2024г.

АКТ
об использовании результатов кандидатской диссертационной работы Мазурка Дмитрия Валерьевича

От лица компании заявляю, что Мазурка Д.В. представил свой алгоритм оценки устойчивости нейросети для нашей системы обнаружения сетевых вторжений, разработанный в его диссертационной работе «Алгоритмы глубокого машинного обучения в системах анализа сетевого трафика». Опробовав такой подход, мы установили, что использование данного подхода к оценке устойчивости снизило деградацию точности со временем в среднем на 10%, а также увеличило сроки на переобучения до 20% и более у эксплуатируемых в промышленной среде моделей за счет отбора наиболее робастных кандидатов при валидации и расширении использования методов регуляризации.

Полученные данные применяются ООО «АйТиАрт» при эксплуатации нейросетевых классификаторов, в том числе с использованием Больших языковых моделей.

Генеральный директор _____ / Варёнова С.В.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования "Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых" (ВлГУ)

«УТВЕРЖДАЮ»
Проректор ВлГУ по образовательной деятельности
_____ А.А. Панфилов
«20» мая 2025 г.

АКТ
о внедрении результатов диссертационных исследований
Мазурка Дмитрия Валерьевича
на кафедре «Информатика и защита информации» ВлГУ

Комиссия в составе: председателя: заведующего кафедрой «Информатика и защита информации» д.т.н., профессора Монахова М.Ю. и членов комиссии: доцента кафедры «Информатика и защита информации» к.т.н., доцента Тельного А.В., доцента кафедры «Информатика и защита информации» к.т.н. Полянского Д.А., доцента кафедры «Информатика и защита информации» к.т.н. Монахова Ю.М. составили настоящий акт о том, что результаты диссертационных исследований Мазурка Дмитрия Валерьевича «Алгоритмы глубокого машинного обучения в системах анализа сетевого трафика» использованы в учебном процессе на кафедре «Информатика и защита информации» ВлГУ. Основные результаты диссертации: алгоритм оценки интерпретируемости результатов работы нейросети, для сравнения различных методов интерпретации решений анализатора сетевого трафика в задачах бинарной классификации редких событий; алгоритм оценки устойчивости нейросети анализатора сетевого трафика, позволяющий оценить пригодность использования классификатора в условиях зашумленности данных; алгоритм интерпретации решений классификатора анализатора сетевого трафика для коррекции ошибочных решений модели, реализованы в ВлГУ при подготовке учебно-методического материала по дисциплине «Методы анализа данных» (программа бакалавриата по направлению 10.03.01 «Информационная безопасность» и программа специалитета по специальности «Информационно-аналитические системы безопасности»), а также апробированы при проведении лабораторных работ в учебных группах ИБ-121, ИСБ-121. Реализация результатов исследования способствовала улучшению качества учебного процесса и повышению профессиональных компетенций обучающихся.

Председатель комиссии
д.т.н., профессор _____ Монахов М.Ю.

Члены комиссии:
к.т.н., доцент _____ Тельный А.В.

к.т.н. _____ Полянский Д.А.

к.т.н. _____ Монахов Ю.М.

ПРИЛОЖЕНИЕ 9

Свидетельства о государственной регистрации интеллектуальной собственности

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2024682642

Модуль оценки значимости по результатам эксперимента

Правообладатель: ***Мазурок Дмитрий Валерьевич (RU)***

Авторы: ***Мазурок Дмитрий Валерьевич (RU), Полховский Владимир Сергеевич (RU)***

Заявка № **2024681285**
Дата поступления **14 сентября 2024 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***25 сентября 2024 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
Сертификат 0692e7e1ab300bf54f240f670bca2026
Владелец Зубов Юрий Сергеевич
Действителен с 10.07.2024 по 03.10.2025

Ю.С. Зубов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2024682916

Программа-API для проведения эксперимента по оценке повышения интерпретируемости решений нейросетевого классификатора

Правообладатели: ***Мазурок Дмитрий Валерьевич (RU), Басов Тихон Денисович (RU), Масляных Анастасия Дмитриевна (RU)***

Авторы: ***Мазурок Дмитрий Валерьевич (RU), Басов Тихон Денисович (RU), Масляных Анастасия Дмитриевна (RU), Гришин Алексей Дмитриевич (RU)***

Заявка № **2024681297**
Дата поступления **15 сентября 2024 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***30 сентября 2024 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
Сертификат 0692e7e1ab300bf54f240f670bcca2026
Владелец Зубов Юрий Сергеевич
Действителен с 10.07.2024 по 03.10.2025

Ю.С. Зубов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2024682393

Программа получения расширенной интерпретации результатов нейросетевого классификатора трафика

Правообладатель: ***Мазурок Дмитрий Валерьевич (RU)***

Автор(ы): ***Мазурок Дмитрий Валерьевич (RU)***

Заявка № **2024681474**
Дата поступления **15 сентября 2024 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***23 сентября 2024 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
Сертификат 0692e761a63005f54f240f670bcca2026
Владелец Зубов Юрий Сергеевич
Действителен с 10.07.2024 по 03.10.2025

Ю.С. Зубов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2021618197

Программный модуль для тестирования скорости работы и точности классификаторов

Правообладатель: ***Федеральное государственное бюджетное образовательное учреждение высшего образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых» (RU)***

Авторы: ***Мазурок Дмитрий Валерьевич (RU), Агафонова Мария Михайловна (RU), Монахов Юрий Михайлович (RU)***

Заявка № **2021617359**
Дата поступления **12 мая 2021 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***24 мая 2021 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2021618155

Программная реализация графического интерфейса для модуля аудита нейронных сетей на подверженность состязательным атакам

Правообладатель: ***Федеральное государственное бюджетное образовательное учреждение высшего образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых» (RU)***

Авторы: ***Мазурок Дмитрий Валерьевич (RU), Монахов Юрий Михайлович (RU), Агафонова Мария Михайловна (RU)***

Заявка № **2021617158**
Дата поступления **12 мая 2021 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***24 мая 2021 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2021618487

Программный модуль оценки актуальности факторов защищенности детектора и классификатора и выдачи адаптивных рекомендаций по повышению защищенности

Правообладатель: ***Федеральное государственное бюджетное образовательное учреждение высшего образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых» (RU)***

Авторы: ***Мазурок Дмитрий Валерьевич (RU), Монахов Юрий Михайлович (RU), Агафонова Мария Михайловна (RU)***

Заявка № **2021617213**
Дата поступления **12 мая 2021 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***27 мая 2021 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2021619081

Программная реализация модуля аудита нейронных сетей на подверженность состязательным атакам

Правообладатель: ***Федеральное государственное бюджетное образовательное учреждение высшего образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых» (RU)***

Авторы: ***Мазурок Дмитрий Валерьевич (RU), Монахов Юрий Михайлович (RU), Агафонова Мария Михайловна (RU)***

Заявка № **2021617231**
Дата поступления **12 мая 2021 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***03 июня 2021 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев