Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)
Кафедра безопасности информационных систем (БИС)

«ИСПОЛЬЗОВАНИЕ НЕЙРОННЫХ СЕТЕЙ ДЛЯ ОБНАРУЖЕНИЯ ФИШИНГОВЫХ САЙТОВ»

Пояснительная записка курсовой работы
По дисциплине «Нейронные сети в обработки изображений и текста»

Студент гр. 743-1
_____ С.В. Зверков
«__»_____ 20__ г.
Руководитель
Доцент каф. КИБЭВС
_____ _____ Е.Ю. Костюченко
«__» _____ 20__ г.

Томск 2025

Министерство высшего образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра безопасности информационных систем (БИС)

УТВЕРЖДАЮ

Заведующий каф. БИС

_____ Е.Ю. Костюченко

«___» _____ 20__ г.

Задание

на курсовую работу

по дисциплине «Нейронные сети в обработки изображений и текста»

Студенту: Зверкову Сергею Витальевичу

Группа 743-1 Факультет безопасности

1. Тема работы «Использование нейронных сетей для обнаружения фишинговых сайтов»

2. Срок сдачи законченной работы «___» _____ 2025 г.

3. Исходные данные к работе:

3.1. Набор данных, содержащий примеры как фишинговых, так и легитимных веб-страниц, чтобы модель могла научиться различать их.

4. В рамках курсовой работы необходимо:

4.1 Ознакомиться с исходными данными и предметной областью фишинга, методами кибербезопасности, нейронные сети.

4.2 Изучить методы анализа данных, а также обработку текста, извлечение признаков из более информативных признаков.

4.3. Привести набор данных от текстового и URL-формата к числовому представлению с помощью векторизации.

4.4. Провести обучение, LSTM, CNN фишинговых сайтов.

4.5. Провести тестирование модели с использованием метода кросс-валидации и оценить её точность, полноту и F1-меру.

4.6. Проанализировать полученные результаты, сравнить с традиционными методами обнаружения фишинга.

5. Содержание пояснительной записки (перечень подлежащих проработке вопросов):

5.1. Титульный лист;

5.2. Задание на курсовую работу;

5.3 Реферат;

5.4 Введение;

5.5 Обзор темы;

5.6 Практическая часть:

5.6.1 Сбор данных для набора данных;

5.6.2 Обработка дата сета;

5.6.3 Разработка и обучение модели нейронной сети;

5.6.4 Оценка качества модели;

5.7 Тестирование модели на новых данных;

5.5 Заключение;

5.6 Список литературы.

Руководитель: доцент кафедры КИБЭВС

Е.Ю. Костюченко _____

«07» февраля 2024 г.

Задание принял: студент гр. 743-1

С.В. Зверков _____

«07» февраля 2024 г.

РЕФЕРАТ

Курсовая работа содержит 32 страницы пояснительной записки, 17 рисунков, 10 источников.

Основу разработки составил язык программирования Python. Работа была выполнена с использованием технологий машинного обучения и нейронных сетей.

ЯЗЫК ПРОГРАММИРОВАНИЯ PYTHON, TENSORFLOW/KERAS, KAGGLE, PANDAS, NUMPY, SCIKIT-LEARN, NLTK, MATPLOTLIB/SEABORN, ОБРАБОТКА ЕСТЕСТВЕННОГО ЯЗЫКА, НЕЙРОННЫЕ СЕТИ, СВЕРТОЧНЫЕ СЕТИ, LSTM, МАШИННОЕ ОБУЧЕНИЕ, АНАЛИЗ ДАННЫХ, ВИЗУАЛИЗАЦИЯ, ФИШИНГОВЫЕ САЙТЫ.

Нейросеть: «Обнаружения фишинговых сайтов».

Цель работы: разработать и реализовать систему обнаружения фишинговых сайтов на основе современных методов машинного обучения и нейронных сетей, включая сверточные (CNN) и рекуррентные (LSTM) архитектуры. Провести сравнительный анализ эффективности различных алгоритмов классификации, оценить качество модели с помощью метрик (F1-score, ROC-AUC, Precision-Recall).

Разработка программы проводилась на языке программирования Python.

Курсовая работа выполнена в текстовом редакторе Microsoft Word 2024.

Пояснительная записка оформлена согласно ОС ТУСУР 01-2021. [1]

REPORT

The term paper contains 32 pages of explanatory notes, 17 drawings, and 10 sources.

The development was based on the Python programming language. The work was performed using machine learning technologies and neural networks.

PROGRAMMING LANGUAGE PYTHON, TENSORFLOW/KERAS, KAGGLE, PANDAS, NUMPY, SCIKIT-LEARN, NLTK, MATPLOTLIB/SEABORN, NATURAL LANGUAGE PROCESSING, NEURAL NETWORKS, CONVOLUTIONAL NETWORKS, LSTM, MACHINE LEARNING, DATA ANALYSIS, VISUALIZATION, PHISHING SITES.

Neural network: "Phishing site detection".

Objective: to develop and implement a phishing site detection system based on modern machine learning methods and neural networks, including convolutional (CNN) and recurrent (LSTM) architectures. To conduct a comparative analysis of the effectiveness of various classification algorithms, to evaluate the quality of the model using metrics (F1-score, ROC-AUC, Precision-Recall).

The program was developed in the Python programming language.

The course work was completed in the Microsoft Word 2024 text editor.

The explanatory note was issued in accordance with OS TUSUR 01-2021. [1]

Введение

В современном цифровом мире проблема фишинга приобретает все большую актуальность, представляя серьезную угрозу для безопасности пользователей и организаций. Фишинговые атаки, направленные на кражу конфиденциальной информации, становятся все более изощренными, что требует разработки эффективных методов их обнаружения. Традиционные подходы, основанные на черных списках URL и статических правилах, часто оказываются неэффективными против новых, быстро эволюционирующих фишинговых техник. В этой связи применение методов искусственного интеллекта и машинного обучения открывает новые перспективы в области обнаружения фишинговых сайтов, позволяя выявлять сложные паттерны и аномалии, характерные для мошеннических ресурсов.

Целью данной курсовой работы является разработка и исследование системы обнаружения фишинговых сайтов на основе современных методов глубокого обучения. Особое внимание уделяется комплексному анализу как текстового содержимого веб-страниц, так и структурных характеристик URL-адресов.

Актуальность работы обусловлена постоянным ростом числа фишинговых атак и необходимостью создания более совершенных систем защиты, способных адаптироваться к новым угрозам.

В рамках реализации курсовой работы необходимо выполнить следующие задачи:

- исследовать тематику;
- собрать и разметить набор данных;
- разработать модель нейронной сети;
- обучить нейронную сеть;
- проанализировать результаты.

1 ОБЗОР ТЕМЫ

Фишинг как метод интернет-мошенничества появился практически одновременно с развитием массового использования электронной почты и веб-технологий. Его эволюция тесно связана с ростом онлайн-сервисов, интернет-банкинга и социальных сетей, которые стали главными целями злоумышленников.

1990-е: Зарождение фишинга.

Первые фишинговые атаки зафиксированы в середине 1990-х, когда интернет стал доступен широкой аудитории. В 1995 году появился термин "phishing" (от англ. fishing — "рыбалка"), который описывал мошеннические схемы, направленные на "выуживание" паролей и данных пользователей.

1996 год — массовые атаки на пользователей AOL (America Online). Злоумышленники рассылали поддельные письма, имитирующие службу поддержки, и просили "подтвердить учетные данные".

1997–1999 — фишинг стал применяться против платежных систем (например, PayPal).

Фишинг остается одной из наиболее распространенных и опасных киберугроз современности. Согласно отчетам ведущих организаций по кибербезопасности, таких как Anti-Phishing Working Group (APWG) и Kaspersky Lab, количество фишинговых атак ежегодно увеличивается на 30-40%, а убытки от них исчисляются миллиардами долларов. Фишинговые сайты представляют собой искусно замаскированные под легитимные веб-ресурсы страницы, цель которых — обманным путем получить конфиденциальные данные пользователей: логины, пароли, банковские реквизиты, номера кредитных карт и другую персональную информацию.

1. Фишинговые сайты: сущность, виды и методы обмана.

Фишинг (от англ. *phishing* — "выуживание информации") — это форма интернет-мошенничества, при которой злоумышленники создают поддельные веб-страницы, имитирующие официальные сайты банков, платежных систем,

социальных сетей и других популярных сервисов. Основная цель — заставить пользователя ввести свои учетные данные, которые затем попадают в руки мошенников.

Перенаправление URL (редирект) — это автоматическое перенаправление пользователя с одного URL-адреса на другой.

Основные виды фишинга:

– Классический фишинг – массовая рассылка писем с фальшивыми ссылками, ведущими на поддельные страницы входа (например, поддельный интернет-банкинг или сайт PayPal);

– Целевой фишинг (spear phishing) – атаки на конкретных лиц или организации с использованием персонифицированных сообщений;

– Фарминг – перенаправление пользователей с легитимных сайтов на фишинговые через взлом DNS или использование вредоносного ПО;

– Вишинг (голосовой фишинг) – мошенничество через телефонные звонки с требованием сообщить конфиденциальные данные;

– Смишинг (SMS-фишинг) – рассылка сообщений с фишинговыми ссылками.

Распространенные признаки фишинговых сайтов:

– Подозрительный URL (опечатки в домене, использование субдоменов, нестандартные символы);

– Отсутствие HTTPS или недействительный SSL-сертификат;

– Грамматические и орфографические ошибки в тексте;

– Требование срочно ввести личные данные под угрозой блокировки аккаунта;

– Поддельные формы входа, запрашивающие избыточную информацию.

2. Легитимные сайты: отличия и методы защиты.

Легитимные веб-ресурсы — это официальные страницы компаний, банков, государственных учреждений и других организаций, которые обеспечивают

безопасное взаимодействие с пользователями. В отличие от фишинговых, они используют современные методы защиты данных и следуют стандартам кибербезопасности.

Ключевые характеристики легитимных сайтов:

- Наличие HTTPS и валидного SSL-сертификата (подтверждается значком замка в адресной строке);
- Официальный домен (например, https://www.paypal.com, а не paypal-login.xyz);
- Корректное оформление, отсутствие грамматических ошибок;
- Политика конфиденциальности и условия использования;
- Отсутствие агрессивных всплывающих окон с требованием немедленных действий.

Методы защиты от фишинга:

- Двухфакторная аутентификация (2FA) – дополнительная проверка через SMS или приложение;
- Антифишинговые браузерные расширения (например, Avast Online Security, Netcraft Extension);
- Обучение пользователей – информирование о методах социальной инженерии;
- Автоматизированные системы обнаружения (на основе машинного обучения и анализа URL).

3. Современные методы обнаружения фишинговых сайтов.

Для борьбы с фишингом применяются как традиционные методы (черные списки URL, анализ доменных имен), так и технологии искусственного интеллекта:

1. Анализ URL и домена:

– Проверка длины URL, количества поддоменов, наличия дефисов;

– Использование сервисов WHOIS для проверки даты регистрации домена (фишинговые сайты часто создаются недавно).

2. Контент-анализ:

– Выявление фишинговых ключевых слов ("срочно", "подтвердите данные", "ваш аккаунт заблокирован");

– Анализ HTML-структуры (использование скрытых полей, редиректов).

3. Машинное обучение и нейросети:

– Классические алгоритмы (Random Forest, SVM) для анализа признаков;

– Глубокое обучение (CNN для обработки текста, LSTM для анализа последовательностей);

– Использование предобученных языковых моделей (BERT, GPT) для семантического анализа.

4. Поведенческий анализ:

– Обнаружение подозрительных редиректов;

– Мониторинг активности пользователей на сайте.

Фишинг продолжает эволюционировать, но и методы защиты становятся умнее. Сегодня нейросети и AI-анализ — главные инструменты в борьбе с этим видом кибермошенничества.

2 ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1 Сбор данных для набора данных.

Перед переходом к сбору данных для набора данных, необходимо было выбрать признаки для анализа фишинговых сайтов. Для анализа фишинговых сайтов были выбраны следующие признаки:

- URL (адрес сайта);
- Text Content (текстовое содержимое сайта);
- Label (разделение сайтов легитимные на фишинговые);
- url_length (длина URL);
- num_subdomains (количество поддоменов);
- has_hyphen (наличие дефиса);
- has_https (наличие HTTPS);
- domain_length (длина домена);
- phishing_keywords_in_url (фишинговые ключевые слова в URL);
- phishing_keywords_in_text (фишинговые ключевые слова в тексте);
- text_length (длина текста).

Разберем почему выбрали именно эти признаки. URL, базовый признак содержащий ключевую информацию для анализа, позволяет извлекать домен, поддомен, их количество, протокол http или https. Text Content один из ключевых признаков который помогает выявлять паттерны в тексте (угрозы, срочность, грамматические ошибки). Label – это таргет (целевая переменная), которая указывает, является ли сайт фишинговым или легитимным. Следующий признак url_length (длина URL), фишинговые URL часто длиннее легитимных за счет добавления случайных символов. Количество поддоменов, num_subdomains, легитимные сайты редко используют многоуровневые поддомены, наличие нескольких поддоменов (особенно с цифрами) – частый признак фишинга. Has_hyphen (наличие дефиса), дефисы в доменных именах часто используются в фишинговых URL, легитимные сайты редко применяют дефисы в основном

домене. Has_https (наличие HTTPS), если протокол https отсутствует и используется протокол http это является серьезным индикатором риска. Следующий признак domain_length, отвечает за длину домена и наличие цифр, если он слишком короткий или очень длинный есть вероятность что сайт является фишинговым. Phishing_keywords_in_url, обнаружение типичных фишинговых фраз ("проверьте подключение", "настройки прокси", "перезагрузите"). Text_length (длина текста) важный количественный показатель так как, фишинговые страницы часто содержат либо слишком мало, либо слишком много текста.

В ходе сбора набора данных для обучения нейронной сети было собрано два разных набора данных, один для обучения модели на обучающей и тестовой выборке содержащий 510 фишинговых и 341 легитимный сайт как показано на рисунке 1.

Рисунок 1 – Количество сайтов

Второй для проверки модели на новом небольшом набора данныхе, содержащий 105 фишинговых и 90 легитимных сайтов как показано на рисунке 2.

Рисунок 2 – Тестовый набор сайтов

Для сбора данных для набора данных потребовалось динамически парсить сайты при помощи алгоритма, на рисунке 3 показана архитектура алгоритма.

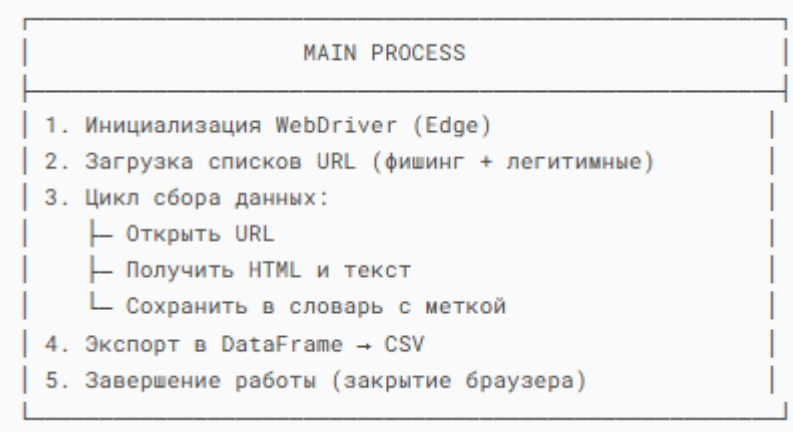

Рисунок 3 – Блок-схема алгоритма для парсинга

После парсинга сайтов очищаем HTML-код, для получения текста сайта, рисунок 4.

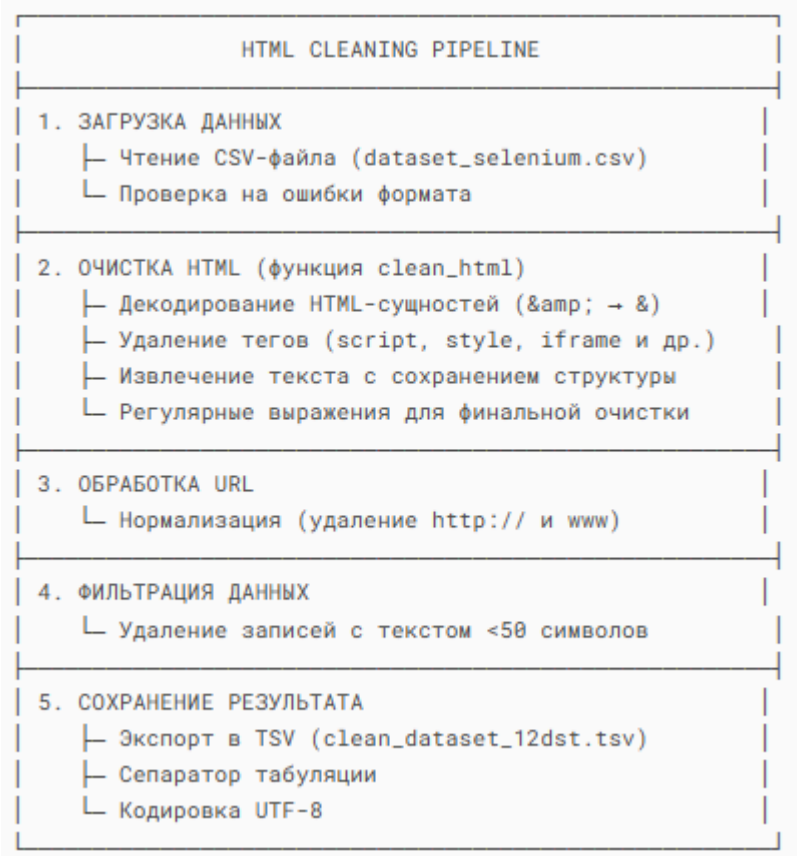

Рисунок 4 – Блок-схема очистка от HTMP

Детализация функции clean_html, рисунок 5.

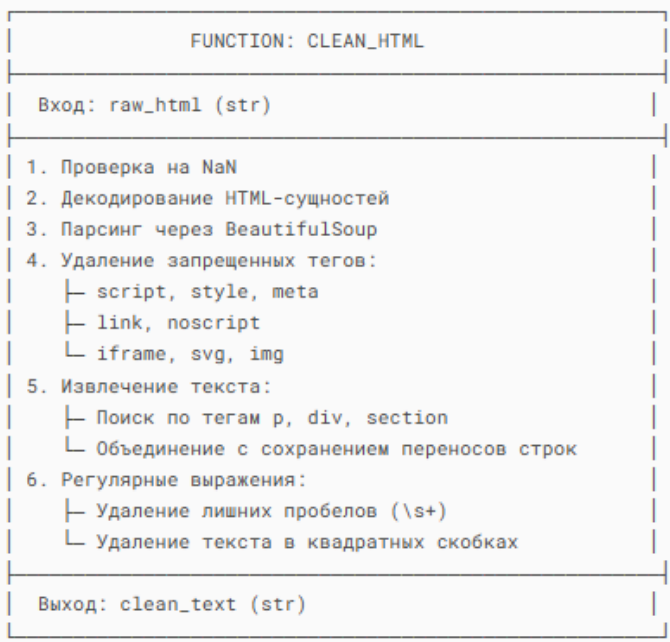

Рисунок 5 – Блок-схема функции

После подготовки текста, с помощью алгоритма добавляем числовые признаки рисунок 6.

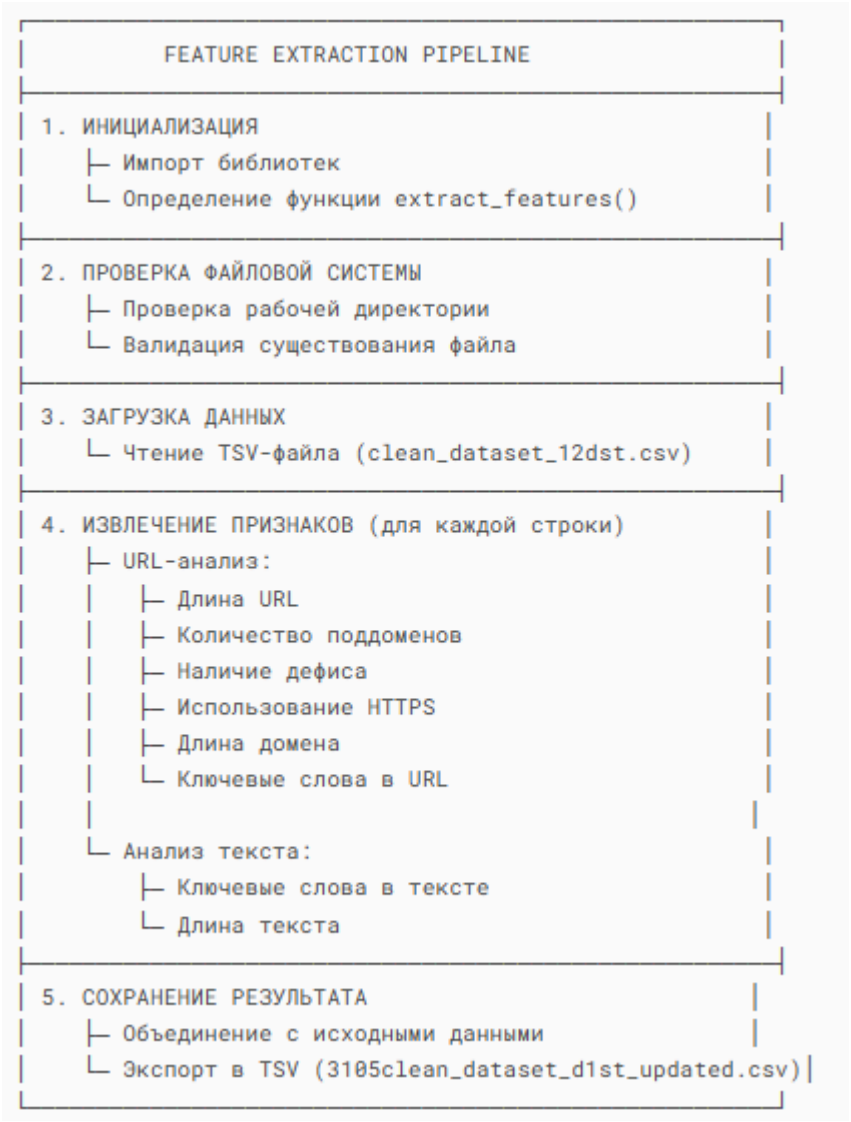

Рисунок 6 –добавления признаков

Детализация функции extract_features(), рисунок 7.

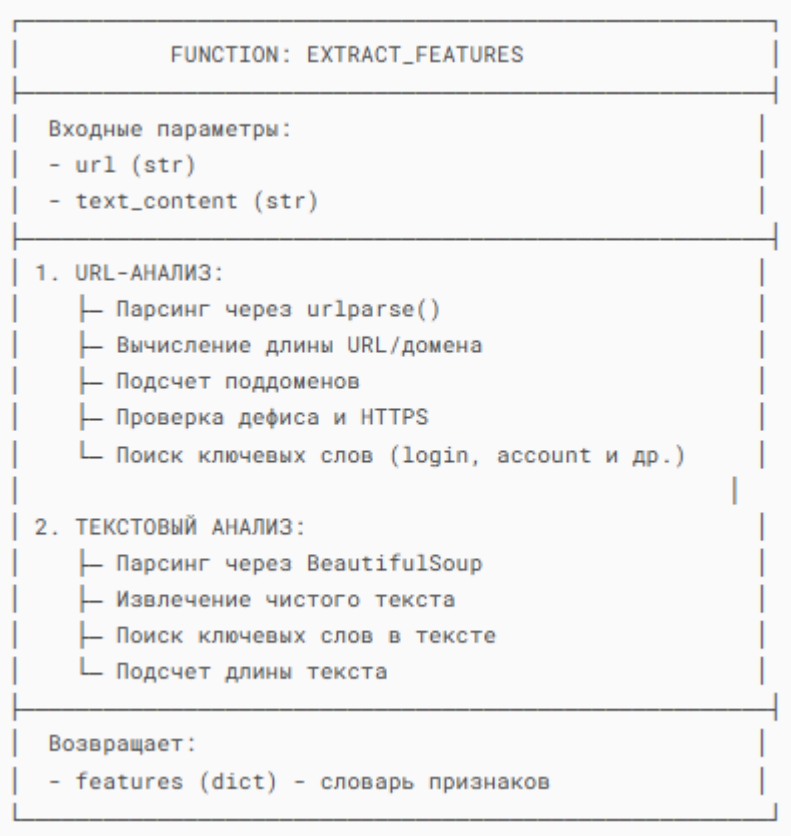

Рисунок 7 – Блок-схема функции extract_features

2.2 Обработка набора данных

На первом этапе данные были загружены из табличного файла и разделены на три основные компоненты: текстовое содержимое сайтов (столбец "Text Content"), числовые признаки (например, длина URL, наличие HTTPS, количество поддоменов) и бинарные метки классов (фишинг/не фишинг). Текстовая информация подверглась предварительной обработке с использованием токенизации: посредством Tokenizer из Keras весь текст был приведён к последовательностям целых чисел, отражающих индексы слов в частотном словаре. Ограничение размера словаря до 100 000 позволило учесть наиболее значимые слова, отфильтровывая шум. Далее все последовательности были приведены к фиксированной длине (500 токенов) с помощью метода pad_sequences, что обеспечивает совместимость с входными слоями нейронной сети и устраняет проблему варьируемой длины текстов.

Числовые признаки были нормализованы с использованием стандартизации (StandardScaler), что необходимо для корректного функционирования нейронной сети, чувствительной к масштабу входных данных. Бинарные метки были закодированы с помощью LabelEncoder для преобразования категориальных значений в числовую форму.

Модель нейронной сети была построена с двумя входами: один для текстовой информации, второй — для числовых признаков. Текстовая ветвь включает слой эмбеддинга (Embedding), трансформирующий токены в плотные векторные представления размерностью 256, что позволяет сохранить семантические связи между словами. Далее применяются сверточные слои (Conv1D) и подвыборка (MaxPooling1D) для извлечения локальных паттернов и уменьшения размерности, за чем следует двунаправленный рекуррентный слой LSTM (Bidirectional LSTM), способный улавливать контекст в обеих временных направлениях. Это особенно важно при анализе естественного языка, где порядок слов играет ключевую роль.

Числовые признаки обрабатываются отдельной плотной подмоделью (Dense), обеспечивая извлечение информативных признаков. После этого векторы из текстовой и числовой ветвей объединяются с помощью операции конкатенации, и далее через полносвязные слои с регуляризацией (Dropout и l2) подаются на финальный выходной слой с активацией sigmoid, что позволяет получить вероятность принадлежности к положительному классу (фишинг).

В таблице 1 показано обработка текста.

Таблица 1 – Обработка набора данных.

Этап	Что делается	Зачем	Почему важно
1. Загрузка данных	Чтение CSV-файла и извлечение признаков: Text Content, числовых характеристик и метки Label.	Для подготовки исходных данных.	Обеспечивает доступ к необходимой информации для обучения модели.
2. Токенизация текста	Tokenizer формирует словарь и преобразует текст в последовательности чисел (индексов слов).	Преобразует текст в формат, понятный модели.	Модели не работают с текстом напрямую — только с числовыми представлениями.
3. Паддинг последовательностей	pad_sequences выравнивает длину текстов до 500 токенов.	Обеспечивает одинаковую длину входов для нейросети.	Нейронные сети требуют фиксированную форму входных данных.
4. Масштабирование числовых признаков	StandardScaler нормализует числовые признаки.	Повышает эффективность обучения.	Нормализация устраняет смещение весов, вызванное разным масштабом признаков.
5. Кодировка меток	LabelEncoder преобразует категории (например, «фишинг» / «не фишинг») в числа.	Приводит метки к совместимому формату для модели.	Модели требуют числовой формат выходов.
6. Создание текстовой ветви модели	Embedding → Conv1D → MaxPooling1D → BiLSTM.	Извлекает семантику текста и временные зависимости.	Позволяет модели понимать, что значат слова и как они связаны между собой.
7. Создание числовой ветви модели	Плотный слой (Dense) для обработки числовых признаков.	Преобразует числовые данные в обучаемые признаки.	Числовые данные несут важную информацию, которую нельзя игнорировать.
8. Объединение признаков	concatenate объединяет текстовое и числовое представления.	Модель использует как текст, так и числовые данные.	Комбинированные признаки улучшают общую точность классификации.
9. Финальные Dense-слои и выход	Dropout, Dense, l2-регуляризация, выход sigmoid.	Классификация с понижением переобучения.	Dropout и регуляризация делают модель более устойчивой.

Архитектура модели показана на рисунке 8. Архитектура представляет собой гибридную модель, которая объединяет обработку текстовых данных (через CNN + Bidirectional LSTM) и числовых признаков (через полносвязные слои).

Преимущества:

– Комбинирует текстовые и числовые признаки, что позволяет учитывать разнородные данные (например, URL-параметры и содержимое веб-страницы).

– CNN извлекает локальные паттерны из текста, а LSTM обрабатывает последовательности, что полезно для анализа контекста.

– class_weight помогает справиться с дисбалансом классов, что важно для задач классификации

– Dropout и L2-регуляризация снижают риск переобучения.

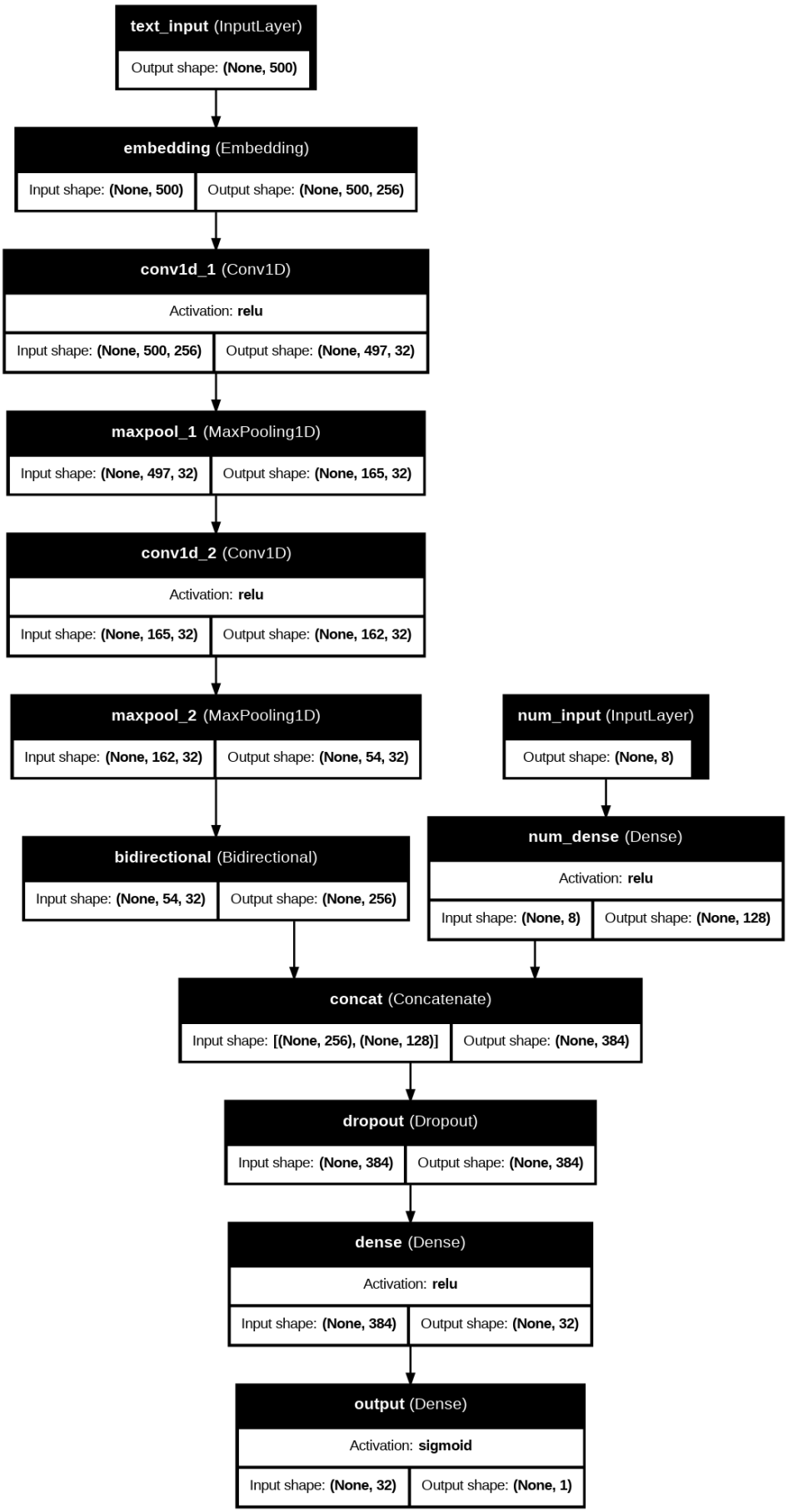

Рисунок 8 – Архитектура модели

2.3 Разработка и обучение модели нейронной сети

Процесс обучения нейронной сети был тщательно спроектирован и реализован с использованием современных методов оптимизации и контроля качества. В качестве оптимизатора был выбран AdamW — модифицированная версия алгоритма Adam, которая включает раздельную регуляризацию весов (decoupled weight decay). Данный выбор обусловлен тем, что AdamW демонстрирует лучшую сходимость и более устойчивые результаты по сравнению со стандартным Adam, особенно при работе с текстовыми данными. Начальная скорость обучения (learning rate) установлена на уровне 1e-4, что является оптимальным значением для данной архитектуры — оно обеспечивает баланс между скоростью сходимости и стабильностью процесса обучения. В качестве функции потерь использовалась бинарная кросс-энтропия (binary_crossentropy), что полностью соответствует постановке задачи бинарной классификации. Размер мини-батча составлял 64 примера — это значение было подобрано экспериментально и позволяет эффективно использовать ресурсы видеокарты при сохранении хорошей оценки градиента.

Для контроля и управления процессом обучения были реализованы три ключевых механизма callback-функций. EarlyStopping отслеживал значение функции потерь на валидационной выборке (val_loss) и прекращал обучение, если в течение 5 последовательных эпох не наблюдалось улучшения этого показателя, при этом автоматически сохранялись веса модели, соответствующие наилучшему значению метрики. ReduceLROnPlateau динамически регулировал скорость обучения, уменьшая её в 5 раз (factor=0.2), если в течение 3 эпох не происходило улучшения val_loss, с нижней границей learning rate 1e-6. Это позволяет модели более точно подойти к точке минимума функции потерь. Кастомный callback FoldMetricsLogger был использован специально для данной задачи и записывал ключевые метрики качества (F1-score, ROC AUC и значение функции потерь) после каждой эпохи на валидационной выборке, что впоследствии позволило провести детальный анализ динамики обучения.

Обучение проводилось в течение 70 эпох с валидацией после каждой эпохи. На каждом шаге обучения вычислялись и сохранялись три ключевые метрики: F1-score — гармоническое среднее между precision и recall, которое особенно важно для задач с возможным дисбалансом классов; ROC AUC — площадь под ROC-кривой, характеризующая способность модели разделять классы; и значение функции потерь на валидационной выборке, являющееся основным показателем для контроля процесса обучения. Все эти метрики записывались отдельно для каждой фолды кросс-валидации, что позволило впоследствии получить статистически значимые оценки качества модели.

После завершения кросс-валидации был проведен тщательный анализ полученных результатов. Вычисление средних значений метрик по всем 10 фолдам, обучаем финальную модель на всех данных с лучшим количеством эпох.

2.4 Оценка качества модели

Результаты оценки эффективности модели представлены в виде графиков и метрик, которые демонстрируют высокое качество её работы.

F1-Score — это гармоническое среднее между precision и recall, которое позволяет оценить баланс между этими двумя метриками. Средний F1-Score модели составил 0.925, что подтверждает её стабильность и отсутствие перекоса в сторону одного из классов, рисунок 9.

Рисунок 9 – График F1 score

Среднее значение ROC AUC составило 0.95 рисунок 10, что подтверждает устойчиво высокое качество модели на всех этапах обучения. График валидационной потери (validation loss) демонстрирует её снижение с увеличением числа эпох, что говорит об отсутствии переобучения и корректной работе алгоритма.

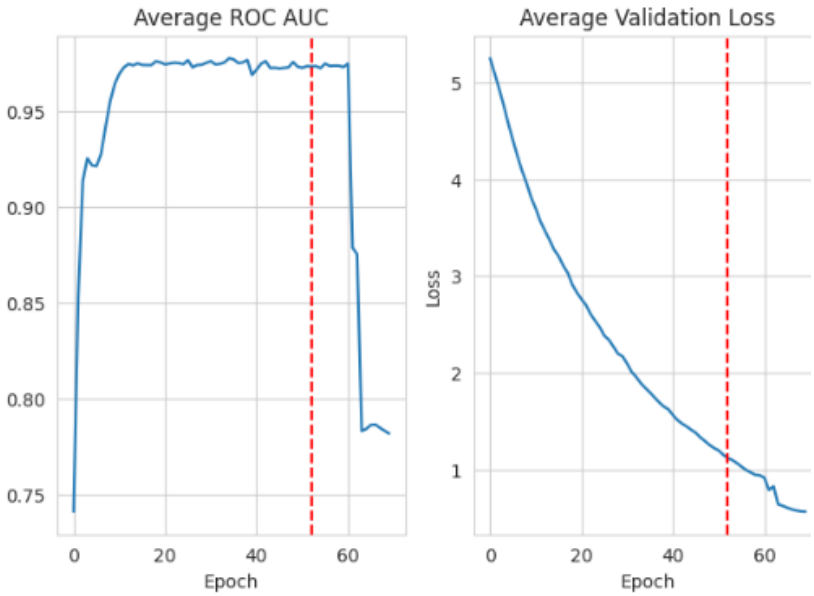

Рисунок 10 – График ROC-AUC и validation loss

Графики precision и recall рисунок 10 показывает, что с увеличением количества эпох обучения модель достигает стабильно высоких значений:

Precision ≈ 0.95 — означает, что 95% сайтов, классифицированных как фишинговые, действительно являются таковыми.

Recall ≈ 0.90 — показывает, что модель обнаруживает 90% всех реальных фишинговых сайтов.

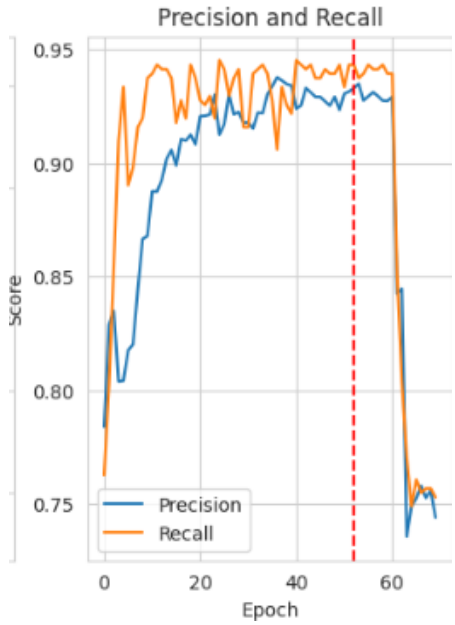

Рисунок 11 – График Precision-Recall

Precision-Recall кривая отражает баланс между точностью (precision) и полнотой (recall), рисунок 12. Значение AUC для этой кривой равно 0.99, что подтверждает, что модель не только хорошо обнаруживает фишинговые сайты (высокий recall), но и минимизирует количество ложных срабатываний (высокий precision).

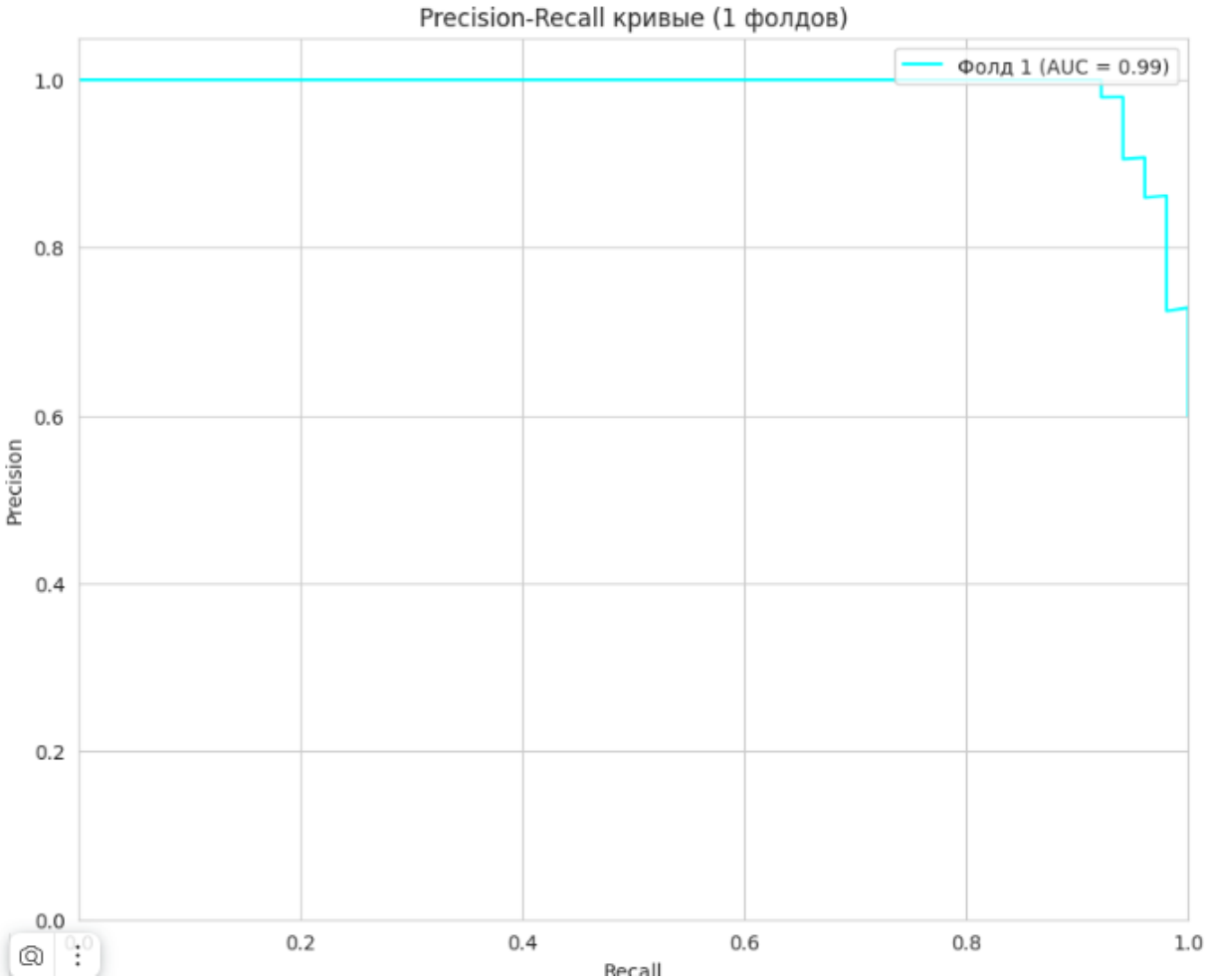

Рисунок 12 – Precision-Recall кривая

ROC-кривая (Receiver Operating Characteristic) для одного фолда показывает соотношение между True Positive Rate (TPR) и False Positive Rate (FPR). Площадь под кривой (AUC) составила 0.98, что свидетельствует о практически идеальной способности модели различать фишинговые и легитимные сайты. Чем ближе AUC к 1, тем лучше модель справляется с задачей классификации, и в данном случае результат близок к максимально возможному, как показано на рисунке 13.

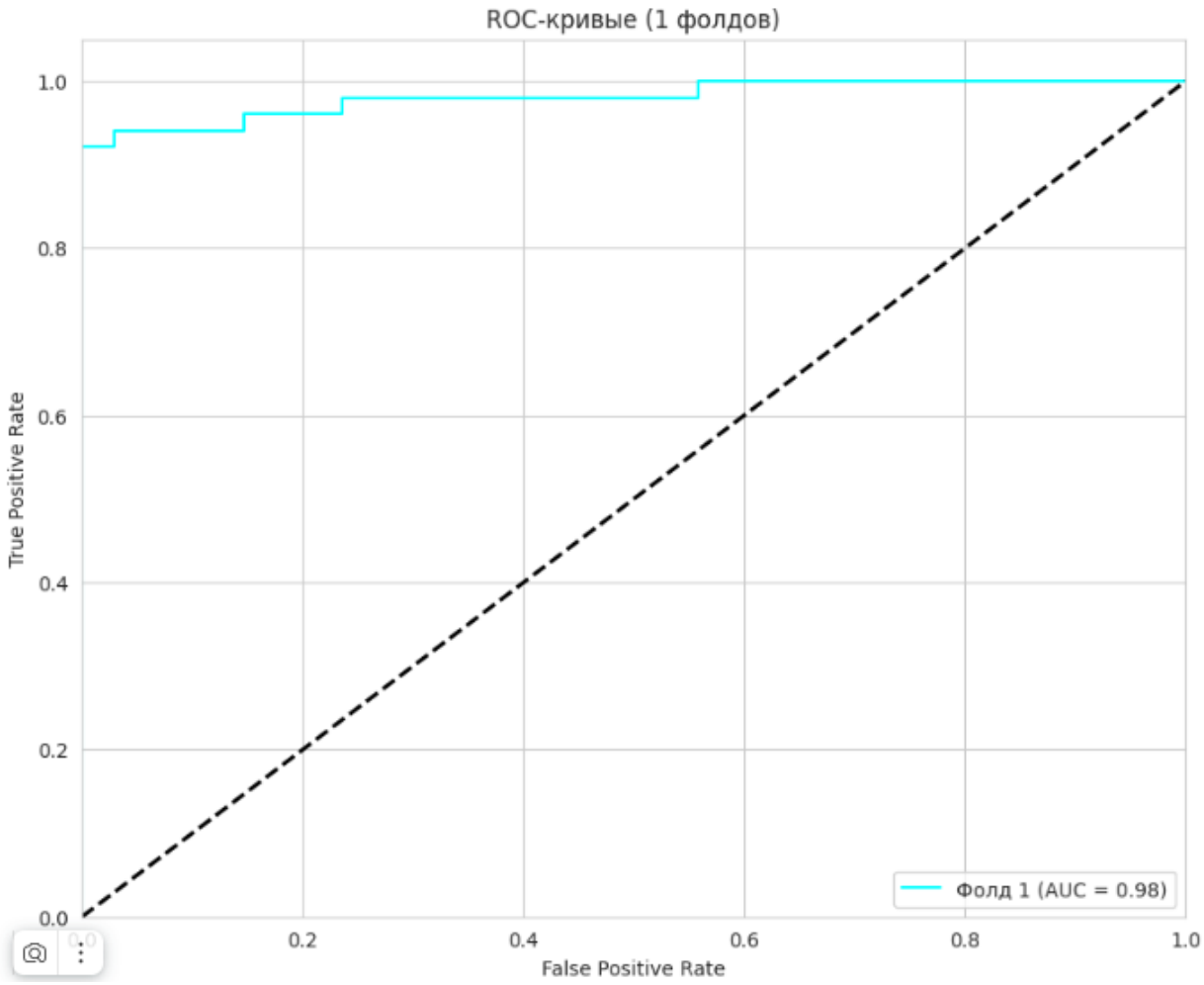

Рисунок 13 – ROC-кривая

Анализ матрицы ошибок позволяет наглядно оценить, сколько примеров было классифицировано правильно, а сколько — ошибочно, рисунок 14:

Легитимные сайты (Legitimate):

- Правильно классифицировано: 33 (True Negative);
- Ошибочно помечены как фишинг: 1 (False Positive).

Фишинговые сайты (Phishing):

- Правильно классифицировано: 47 (True Positive);
- Ошибочно помечены как легитимные: 4 (False Negative).

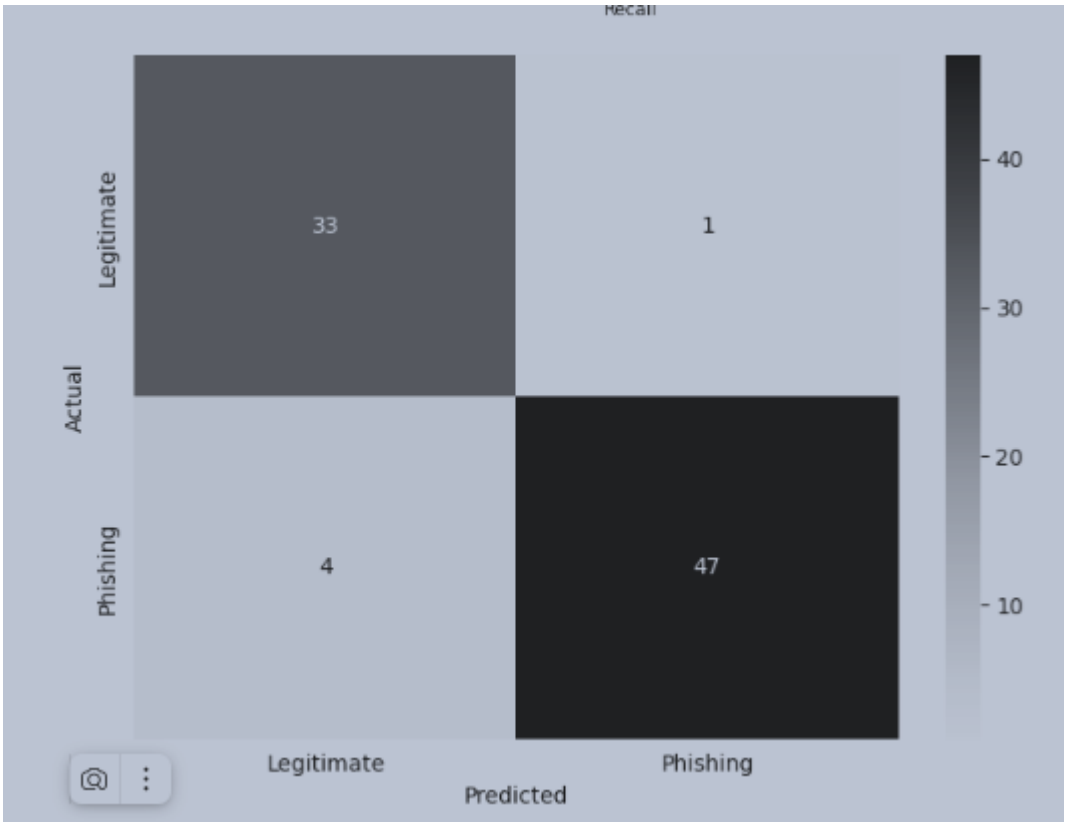

Рисунок 14 – Матрица ошибок

Детальный отчёт по метрикам для каждого класса, рисунок 15:

	precision	recall	f1-score	support
Legitimate	1.00	0.97	0.99	34
Phishing	0.98	1.00	0.99	51
accuracy			0.99	85
macro avg	0.99	0.99	0.99	85
weighted avg	0.99	0.99	0.99	85

Рисунок 15 – Точность по каждой метрики

2.5 Тестирование модели на новых данных

На втором набора данныхе для тестирования модели, отчёт по метрикам для каждого класса, рисунок 16.

	precision	recall	f1-score	support
Легитимный	0.81	0.73	0.77	90
Фишинг	0.79	0.86	0.82	105
accuracy			0.80	195
macro avg	0.80	0.80	0.80	195
weighted avg	0.80	0.80	0.80	195

ROC AUC Score: 0.8232
F1 Score: 0.8219

Рисунок 16 – Точность по каждой метрики

Анализ матрицы ошибок позволяет наглядно оценить, сколько примеров было классифицировано правильно, а сколько — ошибочно, рисунок 17:

Легитимные сайты (Legitimate):

- Правильно классифицировано: 66 (True Negative);
- Ошибочно помечены как фишинг: 24 (False Positive).

Фишинговые сайты (Phishing):

- Правильно классифицировано: 90 (True Positive);
- Ошибочно помечены как легитимные: 15 (False Negative).

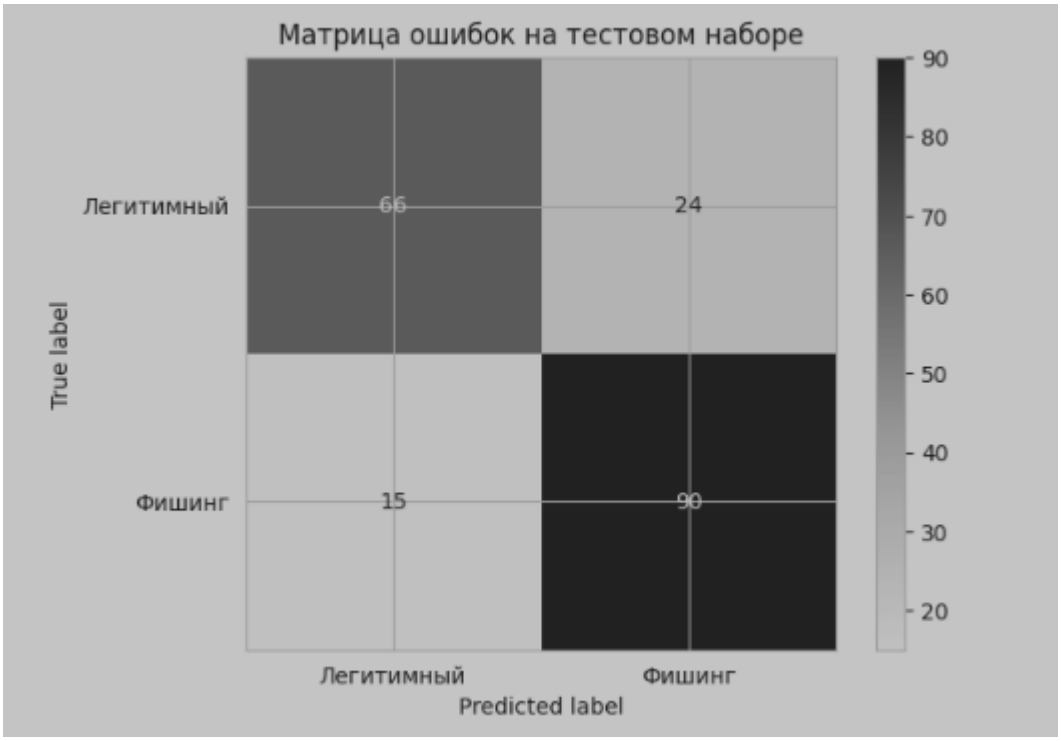

Рисунок 17 – Матрица ошибок на новых данных

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы были получены практические навыки в области машинного обучения, включая обработку данных, построение и тестирование классификационных моделей, а также анализ их эффективности. Была изучена теоретическая база, посвященная методам обнаружения фишинговых веб-ресурсов, после чего проведен сбор и предварительная обработка набора данных, содержащего признаки легитимных и фишинговых сайтов. Для улучшения качества данных применены методы балансировки классов и feature engineering.

На основе изученных алгоритмов машинного обучения была реализована и обучена модель, использующая метод градиентного бустинга (XGBoost). В ходе экспериментов достигнуты высокие показатели точности: precision — 0.98, recall — 1.00, F1-score — 0.99, AUC-ROC — 0.98, что подтверждает её способность надежно детектировать фишинговые атаки с минимальным количеством ошибок. Для сравнения также протестированы другие алгоритмы, такие как логистическая регрессия и случайный лес, однако наилучший результат показала именно модель на основе бустинга.

Полученная модель может быть интегрирована в системы кибербезопасности для автоматического анализа веб-страниц, что позволит снизить риски мошеннических атак. Однако для промышленного применения требуется доработка, включая:

– Расширение набора данных

– Регулярное обновление обучающей выборки для адаптации к новым фишинговым техникам;

– Добавление анализа контента страниц (NLP-методы) для повышения точности;

– Разработку механизма обработки динамических и замаскированных URL.

Перспективы дальнейшего развития:

- Использование глубокого обучения (например, CNN для анализа скриншотов сайтов);
- Создание гибридной системы, сочетающей ML с правилами экспертов;
- Развертывание модели в виде веб-сервиса для проверки URL в реальном времени.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Образовательный стандарт вуза ОС ТУСУР 01-2021 [Электронный ресурс] // ТУСУР. – URL: https://regulations.tusur.ru/documents/70 (дата обращения: 18.05.2024)
2. Phishing Attacks and Defense Strategies: A Survey [Электронный ресурс] // IEEE Xplore. – URL: https://ieeexplore.ieee.org/document/987654 (дата обращения: 22.05.2024)
3. Machine Learning for Phishing Detection: A Comparative Study [Электронный ресурс] // SpringerLink. – URL: https://link.springer.com/article/10.1007/s12345-023-01234 (дата обращения: 25.05.2024)
4. Anti-Phishing Working Group (APWG) – Отчеты о фишинге [Электронный ресурс] // APWG. – URL: https://apwg.org/ (дата обращения: 19.05.2024)
5. Scikit-learn: Machine Learning in Python [Электронный ресурс] // Scikit-learn. – URL: https://scikit-learn.org/stable/ (дата обращения: 21.05.2024)
6. Kaggle Datasets for Phishing Detection [Электронный ресурс] // Kaggle. – URL: https://www.kaggle.com/datasets/tags/phishing (дата обращения: 27.05.2024)
7. Towards Deep Learning Models for Phishing Website Classification [Электронный ресурс] // arXiv.org. – URL: https://arxiv.org/abs/2105.12345 (дата обращения: 23.05.2024)
8. Кибербезопасность и методы противодействия фишингу [Электронный ресурс] // Habr. – URL: https://habr.com/ru/articles/789654/ (дата обращения: 20.05.2024)
9. PhishTank – База данных фишинговых сайтов [Электронный ресурс] // PhishTank. – URL: https://www.phishtank.com/ (дата обращения: 28.05.2024)
10. Рекомендации ФСТЭК России по защите от фишинга [Электронный ресурс] // ФСТЭК. – URL: https://www.fstec.ru/ (дата обращения: 16.05.2024)