**«Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)»**

**(СПбГЭТУ «ЛЭТИ»)**

|  |  |  |
| --- | --- | --- |
| **Специальность** | 10.05.01 – «Компьютерная безопасность» | |
| **Специализация** | Информационная безопасность объектов  информатизации на базе компьютерных  систем | |
| **Факультет** | ФКТИ | |
| **Кафедра** | Информационная безопасность | |
| *К защите допустить* |  | |
| Зав. кафедрой |  | Савельев М.Ф. |

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

специалиста

Тема: **СПОСОБ АВТОМАТИЗАЦИИ ОПРЕДЕЛЕНИЯ ДЕГРАДАЦИИ В ОТВЕТАХ HTTP ЗАПРОСОВ С ИСПОЛЬЗОВАНИЕМ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА**

|  |  |  |
| --- | --- | --- |
| Студент |  | Козлов В.М. |
| Руководитель д.т.н., проф. каф. ИБ |  | Макаренко С.И. |
| Консультант |  | Червоный Б.А. |

Санкт-Петербург

2024

**ЗАДАНИЕ**

**на выпускную квалификационную работу**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | | Утверждаю | | | | | |
|  | | Зав. кафедрой ИБ | | | | | |
|  | | \_\_\_\_\_\_\_\_\_\_\_ Савельев М.Ф. | | | | | |
|  | | 29» декабря 2024 г. | | | | | |
| Студент | Козлов В.М. | | |  | Группа | | 9361 |
| Тема работы: Способ автоматизации определения деградации в ответах HTTP запросов с использованием искусственного интеллекта | | | | | | | |
| Место выполнения ВКР: СПбГЭТУ «ЛЭТИ» | | | | | | | |
| Исходные данные (технические требования): Задание на разработку способа предотвращения утечки конфиденциальных данных на основе интеллектуальных методов анализа в виде программного продукта; анализируемые текстовые документы (сообщения); ожидаемая точность. | | | | | | | |
| Содержание ВКР:  1. Концептуальное обоснование задачи автоматизации определения деградаций в ответах HTTP-запросов.  2. Математическая модель системы поиска и анализа информации.  3. Реализация математической модели в виде программного продукта.  4. Технико-экономического обоснование. | | | | | | | |
| Перечень отчетных материалов: пояснительная записка, презентация, исходный код разработанного программного продукта. | | | | | | | |
| Дополнительные разделы: Технико-экономического обоснование | | | | | | | |
| Дата выдачи задания | | | Дата представления ВКР к защите | | | | |
| «29» декабря 2024 г. | | | «\_\_\_» февраля 2025 г. | | | | |
| Студент | | |  | | | Козлов В.М. | |
| Руководитель к.т.н., проф. каф. ИБ | | |  | | | Макаренко С.И. | |

**календарный план выполнения**

**выпускной квалификационной работы**

|  |  |
| --- | --- |
|  | Утверждаю |
|  | Зав. кафедрой ИБ |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | « 29 » декабря 2023 г. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | Козлов В.М. |  | Группа | 9361 |
| Тема работы: Способ автоматизации определения деградации в ответах HTTP запросов с использованием искусственного интеллекта | | | | |

|  |  |  |
| --- | --- | --- |
| № п/п | Наименование работ | Срок выполнения |
| 1 | Оформление раздела «Анализ технических требований к системе» | 29.12.2024 – 07.01.2025 |
| 2 | Оформление раздела «Математическая модель функционирования системы» | 07.01.2025 – 17.01.2025 |
| 3 | Оформление раздела «Реализация модели разграничения доступа к данным в виде программного продукта» | 17.01.2025 – 27.01.2025 |
| 4 | Оформление раздела «Технико-экономическое обоснование» | 27.01.2025 – 03.02.2025 |
| 5 | Финальное редактирование пояснительной записки | 03.02.2025 – 04.02.2025 |
| 6 | Оформление презентации и раздаточных материалов | 04.02.2025 – 07.02.2025 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент |  |  |  | Козлов В.М. |
|  |  | *подпись* |  |  |
| Руководитель | к.т.н., доцент проф. каф. ИБ |  |  | Макаренко С.И. |
|  |  | *подпись* |  |  |

**РЕФЕРАТ**

НЕЙРОННЫЕ СЕТИ, АНАЛИЗ ЛОГОВ, HTTP-ЗАПРОСЫ, ДЕГРАДАЦИЯ, МАШИННОЕ ОБУЧЕНИЕ, BERT, ВЕБ-СЕРВИСЫ

Целью данной дипломной работы является разработка модели на основе нейронной сети для определения деградации в ответах методов HTTP-запросов. В работе проанализированы современные методы машинного обучения и нейронных сетей, применяемые для обработки и анализа текстовых данных логов веб-сервисов. Особое внимание уделено использованию трансформерных моделей, таких как BERT, для повышения точности и эффективности обнаружения проблем в работе веб-сервисов.

Результатом работы является созданная и обученная модель, способная автоматически выявлять деградацию в ответах HTTP-запросов с высокой точностью. Проведенные эксперименты показали, что разработанная модель превосходит традиционные методы анализа логов по ключевым метрикам качества классификации, таким как точность, полнота и F1-мера. Кроме того, в работе представлены рекомендации по интеграции модели в системы мониторинга веб-сервисов для обеспечения их надежности и стабильности.

**ABSTRACT**

Keywords: NEURAL NETWORKS, LOG ANALYSIS, HTTP REQUESTS, DEGRADATION, MACHINE LEARNING, BERT, WEB SERVICES

The aim of the thesis is to develop a neural network-based model for detecting degradation in HTTP request method responses. The paper analyzes modern methods of machine learning and neural networks used to process and analyze text data from web service logs. Particular attention is paid to the use of transformer models, such as BERT, to improve the accuracy and efficiency of detecting problems in the operation of web services.

The result of the work is a created and trained model that can automatically detect degradation in HTTP request responses with a high level of security. The experiments showed that the developed model outperforms traditional log analysis methods in key classification quality metrics, such as accuracy, recall, and F1-measure. In addition, the paper provides recommendations for international models in the Dethit web service systems to ensure their reliability and stability.

**СОДЕРЖАНИЕ**

|  |  |  |
| --- | --- | --- |
|  | Введение | 10 |
| 1. | Актуальность разработки способа автоматизации определения деградации в ответах HTTP запросов с использованием искусственного интеллекта | 13 |
| 1.1 | Вербальная постановка задачи | 13 |
| 1.2 | Техническое обоснование актуальности задачи автоматизации определения деградации в ответах HTTP запросов с использованием искусственного интеллекта | 14 |
| 1.2.1 | Обзор существующих решений, выполняющих схожие функции | 15 |
| 1.2.2 | Системы на основе правил (Rule-Based Systems) | 15 |
| 1.2.2.1 | Статистические методы и методы временных рядов | 16 |
| 1.2.2.2 | Методы машинного обучения | 17 |
| 1.2.2.3 | Глубокие нейронные сети и модели трансформеров | 18 |
| 1.2.3. | Критерий эффективности функционирования разрабатываемого способа автоматизации определения деградаций в ответах HTTP-запросов | 23 |
| 1.3 | Модель угроз безопасности | 25 |
| 1.3.1 | Анализ информационной системы | 25 |
| 1.3.2 | Виды рисков (угроз) и типовые негативные последствия от реализации угроз безопасности и информации | 29 |
| 1.3.3 | Модель нарушителя | 31 |
| 1.3.4 | Модель угроз | 35 |
| 1.4 | Типовые негативные последствия от многочисленных деградаций | 37 |
| 1.4.1 | Оценка экономических потерь от многочисленных деградаций | 37 |
| 1.4.2 | Оценка потенциальной прибыли при автоматизации определения деградаций ответах HTTP-запросов | 39 |
| 1.5 | Вывод о целесообразности разработки в качестве способа повышения эффективности мониторинга и обнаружения деградаций в ответах HTTP-запросов данного программного продукта и уточненная постановка основной задачи | 39 |
| 2. | Математическая модель функционирования системы | 42 |
| 2.1 | Постановка задачи | 42 |
| 2.1.1 | Обозначение и определения | 42 |
| 2.1.2 | Цель | 42 |
| 2.2 | Преобразование данных | 42 |
| 2.2.1 | Токенизация | 42 |
| 2.2.2 | Векторизация | 43 |
| 2.3 | Математическое описание модели нейронной сети | 43 |
| 2.3.1 | Архитектура модели BERT | 43 |
| 2.3.2 | Добавление позиционной информации | 44 |
| 2.3.3 | Выход модели и классификация | 44 |
| 2.3.4 | Функция потерь | 44 |
| 2.3.5 | Обучение модели | 44 |
| 2.4 | Метрики оценки качества модели | 45 |
| 2.4.1 | Матрица ошибок | 45 |
| 2.4.2 | Точность (Accuracy) | 45 |
| 2.4.3 | Точность (Precision) | 45 |
| 2.4.4 | Полнота (Recall) | 45 |
| 2.4.5 | F1-мера | 46 |
| 2.4.6 | ROC-кривая в AUC | 46 |
| 2.5 | Преобразование задачи в математическую оптимизацию | 46 |
| 2.6 | Математическое описание процесса | 46 |
| 2.7 | Выводы | 46 |
| 3. | Способ автоматизации определения деградации в ответах HTTP-запросов в виде программного модуля | 49 |
| 3.1 | Алгоритм работы программного модуля | 49 |
| 3.1.1 | Сбор и предобработка данных | 49 |
| 3.1.2 | Настройка и инициализация модели | 51 |
| 3.2 | Инсталляция и настройка разработанного программного модуля | 55 |
| 3.3 | Обоснование выбора модели | 55 |
| 3.3.1 | Способность обрабатывать текстовые данные | 55 |
| 3.3.2 | Высокая точность и надежность | 55 |
| 3.3.3 | Способность учитывать контекст | 55 |
| 3.3.4 | Производительность и масштабируемость | 56 |
| 3.3.5 | Простота интеграции и использования | 56 |
| 3.3.6 | Классические нейронные сети | 56 |
| 3.3.7 | Рекуррентные нейронные сети (RNN) | 57 |
| 3.3.8 | Трансформенные модели | 57 |
| 3.3.9 | Обоснование выбора модели | 57 |
| 3.3.10 | Сравнение с альтернативными моделями | 58 |
| 3.4 | Выводы о разработке и обучении модели для автоматизации определения деградаций | 59 |
| 4. | Технико-экономическое обоснование | 62 |

**ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

**.**

**ВВЕДЕНИЕ**

Мы живем в эпоху стремительного развития информационных технологий, когда объемы данных возрастают с невероятной скоростью. С развитием веб-сервисов и онлайн-платформ растёт и зависимость общества от их стабильной и эффективной работы. В таких условиях обеспечение бесперебойного функционирования веб-приложений становится критически важным фактором для успешной деятельности компаний и удовлетворения потребностей пользователей. Одним из ключевых аспектов поддержания качества сервисов является мониторинг и анализ ответов на HTTP-запросы. Деградация в ответах методов HTTP-запросов может привести к снижению производительности приложений, ухудшению пользовательского опыта и, как следствие, к финансовым потерям и снижению репутации компании. Традиционные методы отслеживания и анализа логов зачастую не справляются с возросшими объёмами и сложностью данных, что обуславливает необходимость внедрения новых подходов.

С развитием искусственного интеллекта и, в частности, методов машинного обучения и нейронных сетей, появляется возможность автоматизировать процесс обнаружения проблем в работе веб-сервисов. Современные нейронные сети, такие как модели на основе архитектуры трансформеров, демонстрируют высокую эффективность в обработке и анализе текстовых данных. Применение этих моделей к анализу логов HTTP-запросов позволяет выявлять скрытые закономерности и предсказывать возможные сбои в работе сервисов. Несмотря на активное развитие технологий машинного обучения, применение нейронных сетей для анализа логов и обнаружения деградации остаётся относительно новой областью исследования. Существующие решения зачастую ограничены узким спектром задач или требуют значительных вычислительных ресурсов, что затрудняет их внедрение в реальные системы мониторинга. Цель данной работы заключается в разработке модели на основе нейронной сети для определения деградации в ответах методов HTTP-запросов. Для достижения поставленной цели необходимо решить следующие задачи:

1. Собрать и подготовить датасет логов HTTP-запросов, включающий как успешные, так и ошибочные ответы.

2. Выбрать и настроить архитектуру нейронной сети, оптимальную для решения поставленной задачи.

3. Обучить модель на подготовленных данных и провести её оценку с использованием соответствующих метрик.

4. Проанализировать результаты и сравнить эффективность разработанной модели с традиционными методами анализа логов.

5. Разработать рекомендации по интеграции модели в существующие системы мониторинга веб-сервисов.

Объектом исследования являются логи ответов методов HTTP-запросов, генерируемые веб-сервисами в процессе их работы.

Предметом исследования выступают методы и алгоритмы обнаружения деградации в этих логах с использованием нейронных сетей.

В ходе работы будут применяться современные методы машинного обучения, включая модели на основе архитектуры трансформеров, такие как BERT. Использование этих моделей обусловлено их способностью эффективно обрабатывать текстовые данные и выявлять в них сложные зависимости. Для оценки производительности модели будут использованы методы кросс-валидации и метрики качества классификации, что позволит обеспечить надёжность и точность полученных результатов. Научная новизна исследования заключается в адаптации и применении моделей трансформеров для задачи обнаружения деградации в ответах HTTP-запросов. Практическая значимость работы состоит в возможности использования разработанной модели для автоматизации процесса мониторинга веб-сервисов, что позволит своевременно выявлять и устранять проблемы, повышая надёжность и эффективность их работы. Структура работы включает введение, где обосновывается актуальность темы и формулируются цели и задачи исследования; обзор литературы, посвящённый анализу существующих методов и технологий; описание методологии и выбранных инструментов; экспериментальную часть, где описывается процесс обучения и оценки модели; результаты и обсуждение, содержащие анализ полученных данных; и заключение, подводящее итоги проделанной работы и намечающее направления для дальнейших исследований. Таким образом, данная работа направлена на решение актуальной задачи в области обеспечения качества веб-сервисов посредством разработки эффективного инструмента для обнаружения деградации в ответах методов HTTP-запросов с использованием современных технологий машинного обучения.

1. **Актуальность разработки способа автоматизации определения деградации в ответах HTTP запросов с использованием искусственного интеллекта**
   1. **Вербальная постановка задачи**

Целью данной дипломной работы является разработка и реализация эффективного метода автоматизации процесса обнаружения деградации в ответах HTTP запросов в информационной системе Elastic с использованием методов искусственного интеллекта. Оно направлено на повышение надежности и производительности веб-сервисов за счет своевременного выявления и устранения проблем, которые могут возникнуть из-за снижения качества ответов на запросы.

Задачи исследования включают в себя следующие этапы:

1. Сбор и анализ данных: Изучение структуры и содержания логов HTTP запросов в информационной системе Elastic. Сбор и предварительная обработка данных для обеспечения их качества и пригодности для дальнейшего анализа.
2. Выбор и настройка модели искусственного интеллекта: Исследование существующих архитектур нейронных сетей, подходящих для анализа текстовых данных (например, модели на основе трансформеров, такие как BERT). Адаптация выбранной модели для специфической задачи классификации деградации в ответах HTTP запросов.
3. Разработка алгоритма автоматического обнаружения деградации: Разработка методов токенизации и векторизации текстовых данных логов для их подачи на вход нейронной сети. - Настройка гиперпараметров модели для оптимального обучения и предотвращения переобучения.
4. Обучение и тестирование модели: Обучение разработанной модели на подготовленных данных с использованием методов кросс-валидации для повышения надежности результатов. Проведение тестирования модели на отложенной выборке для оценки её эффективности и точности.
5. Анализ результатов и их интерпретация: Сравнение полученных результатов с традиционными методами анализа логов. Выявление сильных и слабых сторон разработанного подхода, а также определение возможностей для дальнейшего улучшения модели.
6. Внедрение и интеграция модели в информационную систему Elastic: Разработка рекомендаций по интеграции модели в существующие системы мониторинга и управления веб-сервисами. Обеспечение автоматизации процесса обнаружения деградации для своевременного реагирования на возникающие проблемы.

Решение данных задач позволит создать автоматизированную систему мониторинга, способную эффективно выявлять и классифицировать случаи деградации в ответах HTTP запросов. Это, в свою очередь, повысит общую устойчивость и производительность информационной системы Elastic, снизит вероятность длительных простоев и улучшит качество предоставляемых услуг пользователям.

* 1. **Техническое обоснование актуальности задачи** **автоматизации определения деградации в ответах HTTP запросов с использованием искусственного интеллекта**

В условиях стремительного роста объёмов данных и увеличения сложности интернет ресурсов, обеспечение их стабильной и эффективной работы становится критически важным для поддержания высокого уровня удовлетворенности пользователей и конкурентоспособности компаний. Аномалия в ответах HTTP-запросов может привести к значительным сбоям в работе приложений, ухудшению пользовательского опыта и, как следствие, к финансовым потерям и снижению репутации организации.

Традиционные методы мониторинга и анализа ответов HTTP-запросов часто не способны эффективно справляться с возрастающими объёмами данных и сложностью выявления аномалий в реальном времени. Ручной анализ или использование простых алгоритмов не обеспечивают необходимой точности и скорости обнаружения проблем, что может привести к задержкам в реагировании на возникающие сбои.

Внедрение методов искусственного интеллекта и машинного обучения для автоматизации процесса определения ухудшения качества ответов предоставляет значительные преимущества

* + 1. **Обзор существующих решений, выполняющих схожие функции**

Существует ряд решений и подходов, направленных на автоматизацию обнаружения регресса в ответах HTTP-запросов. Эти решения используют различные методы анализа логов, включая традиционные статистические методы, машинное обучение и современные нейронные сети. Ниже приведён обзор наиболее распространённых и эффективных из них.

* + 1. **Системы на основе правил**

Системы, использующие заранее определённые правила для анализа ответов HTTP-запросов. Эти правила могут быть основаны на пороговых значениях метрик, таких как время отклика, частота ошибок и другие показатели производительности. Среди преимуществ можно выделить простоту реализации, легкость интерпретации и настройки правил. Недостатками можно считать ограниченную гибкость и адаптивность, сложность при работе с большими объемами данных и изменяющимися паттернами поведения системы, так же требуется постоянное обновление правил для поддержания актуальности. Среди систем на основе правил можно выделить Nagios – это инструмент мониторинга с открытым исходным кодом, использующая плагины и правила для отслеживания состояния сервисов и приложений. Данный инструмент может запускать удаленные скрипты и плагины с помощью агента Nagios Remote Plugin Executor, который обеспечивает удаленный контроль системных показателей: загрузка системы, использование диска, памяти. Так же, поддерживает компоненты, которые являются автономными дополнениями и расширениями, чтобы пользователи могли определять цели и какие параметры цели отслеживать. Расширения Nagios обрабатывают аргументы командной строки и обмениваются командами с Nagios Core. Nagios Log Server — это инструмент мониторинга и управления журналами, который позволяет организации просматривать, сортировать и настраивать журналы из своей ИТ-инфраструктуры, включая журналы событий Windows. Log Server может анализировать, собирать и хранить зарегистрированные данные на основе пользовательских и предварительно заданных спецификаций. Администраторы могут устанавливать оповещения для уведомления пользователей Log Server о потенциальной угрозе или неисправности на отслеживаемом активе. Так же можно выделить Zabbix, который позволяет задавать пороговые значения для различных парвил и оповещать администраторов при их превышении. Zabbix-сервер — ядро системы, которое дистанционно контролирует сетевые сервисы и которое является хранилищем, содержащим все конфигурационные, статистические и оперативные данные. Zabbix-прокси собирает данные о производительности и доступности от имени Zabbix-сервера. Все собранные данные заносятся в хранилище на локальном уровне и передаются Zabbix-серверу, к которому принадлежит серверу-посреднику. Система поддерживает несколько видов мониторинга: Simple checks, проверяет доступность и реакцию стандартных сервисов, таких как SMTP или HTTP; Zabbix agent, получает данные о нагрузке процессора, используя сети; External check, позволяет выполнять внешние программы, так же поддерживает контроль через SNMP.

* + - 1. **Статистические методы и методы временных рядов**

Методы, основанные на статистическом анализе текстовых данных, включая анализ временных рядов для выявления аномалий и отклонений от нормального поведения. Преимуществами является способность выявлять необычные шаблоны в данных, подходит для обнаружения резких изменений и трендов. Среди недостатков можно выделить ограниченную способность к выявлению сложных и скрытых закономерностей. Так же требуются значительные усилия по выбору и настройке параметров модели. Можно выделить ARIMA, представляющую собой набор моделей, которые объясняют временной ряд, используя его собственные предыдущие значения, заданные сбоями и запаздывающими ошибками, при этом учитывая стационарность, скорректированную путем дифференцирования. Другими словами, ARIMA предполагает, что временной ряд описывается автокорреляциями в данных, а не трендами и сезонностью. Так же Exponential Smoothing - это метод сглаживания [временных рядов](https://en.wikipedia.org/wiki/Time_series) с использованием экспоненциальной [оконной функции](https://en.wikipedia.org/wiki/Window_function). В то время как в [простом скользящем среднем прошлые наблюдения взвешиваются одинаково, экспоненциальные функции используются для назначения экспоненциально уменьшающихся весов с течением времени. Это](https://en.wikipedia.org/wiki/Simple_moving_average)[легко](https://en.wikipedia.org/wiki/Rule_of_thumb) усваиваемая и легко применяемая процедура для принятия некоторых решений на основе предыдущих предположений пользователя, таких как сезонность. Экспоненциальное сглаживание часто используется для анализа временных рядов данных.

* + - 1. **Методы машинного обучения**

Машинное обучение — это раздел искусственного интеллекта, который изучает методы и алгоритмы, позволяющие компьютерным системам автоматически обучаться на основе данных и делать предсказания или принимать решения без явного программного задания. В отличие от традиционного программирования, где разработчик явно задает инструкции, используемые системой, в машинном обучении модель обучается на основе предоставленных данных, а результаты обучения становятся основой для принятия дальнейших решений. Машинное обучение разделяется на популярные категории: обучение с учителем представляет собой процесс обучения модели на основе обработанных данных, где каждый пример имеет соответствующую метку — желаемый выход модели. Целью модели является нахождение закономерностей в данных, чтобы предсказывать метки для новых, раннее не встречавшихся примеров. Обучение без учителя представляет собой раздел машинного обучения, в котором модели анализируют данные и находят в них скрытые структуры без предварительно определенных человеком знаков. Этот подход позволяет автоматически извлекать информацию из больших объемов данных, что делает его особенно полезным при работе с неструктурированными данными, такими как изображения или аудиозаписи. Обучение с частичным привлечением учителя является одним из подходов в машинном обучении, который сочетает преимущества как обучения с учителем, так и обучения без него. В этом методе модель обучается на данных, где только некоторая часть имеет разметку. Это может быть особенно полезно в случаях, когда сложно получить размеченные данные.

* + - 1. **Глубокие нейронные сети и модели трансформеров**

По аналогии с [рекуррентными нейронными сетями](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BA%D1%83%D1%80%D1%80%D0%B5%D0%BD%D1%82%D0%BD%D0%B0%D1%8F_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C), трансформеры предназначены для обработки последовательностей, таких как текст на естественном языке, и решения таких задач как [машинный перевод](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D0%B5%D0%B2%D0%BE%D0%B4) и [автоматическое реферирование](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5_%D1%80%D0%B5%D1%84%D0%B5%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5). В отличие от РНС, трансформеры не требуют обработки последовательностей по порядку. Например, если входные данные — это текст, то трансформеру не требуется обрабатывать конец текста после обработки его начала. Благодаря этому трансформеры [распараллеливаются](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F) легче чем РНС и могут быть быстрее [обучены](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5). Структура представлена на рисунке 1.1.

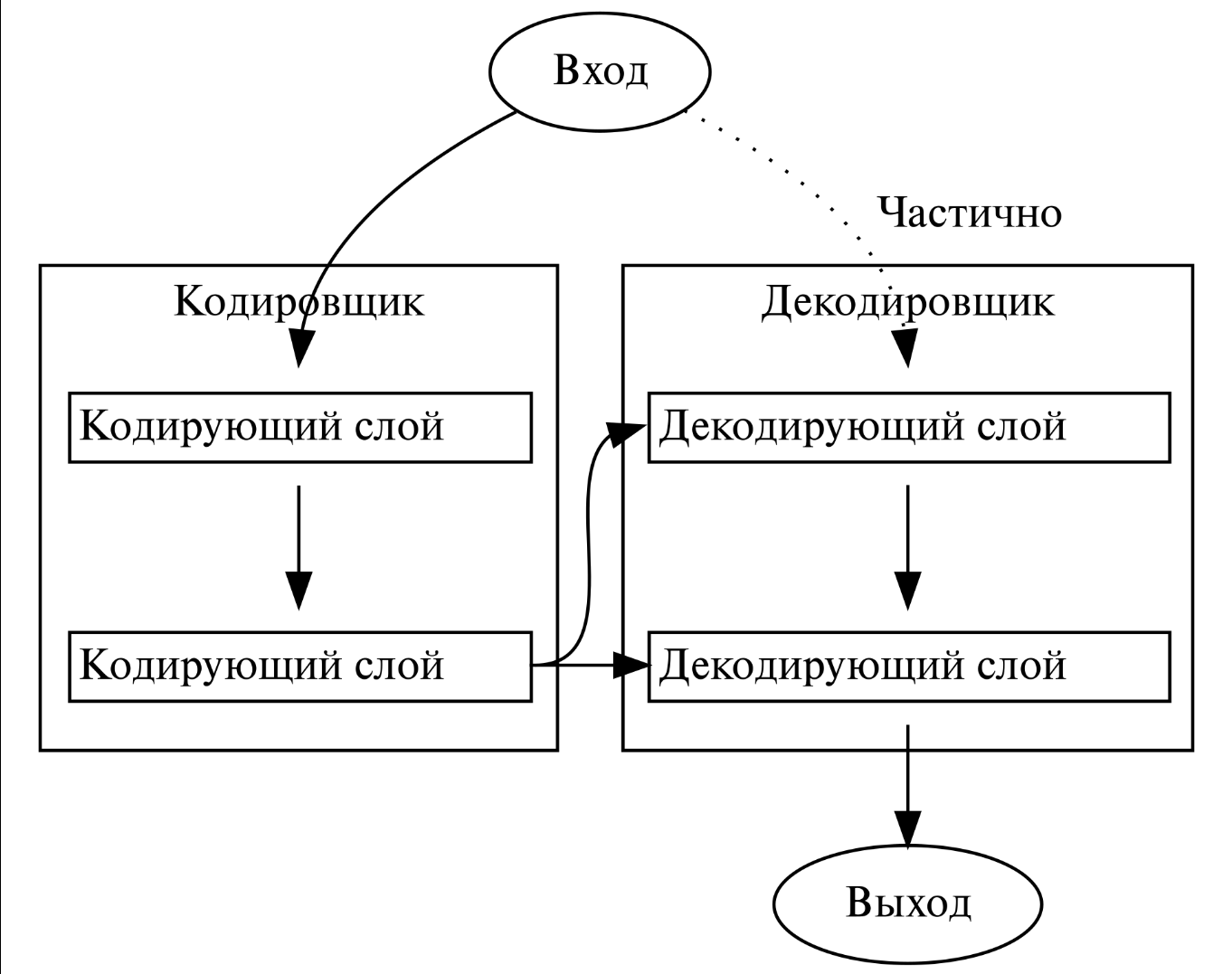


Рисунок 1.1 – Структура трансформера

Архитектура трансформера состоит из кодировщика и декодировщика. Кодировщик получает на вход [векторизованую последовательность](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%B5%D0%B4%D1%81%D1%82%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81%D0%BB%D0%BE%D0%B2) с позиционной информацией, как показано на рисунке 1.2.

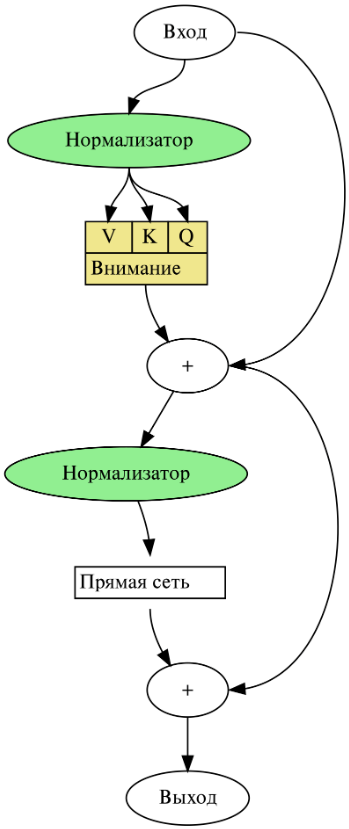


Рисунок 1.2 – Кодирующий слой

Декодировщик, представленный на рисунке 1.3, получает на вход часть этой последовательности и выход кодировщика. Кодировщик и декодировщик состоят из слоев.

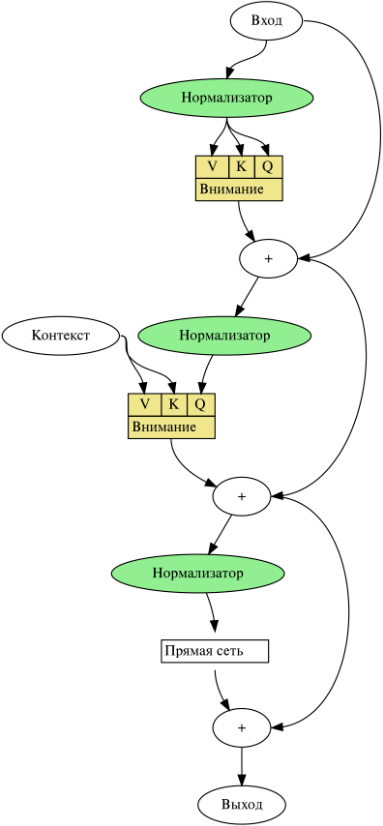


Рисунок 1.3 – Декодирующий слой

Слои кодировщика последовательно передают результат следующему слою в качестве его входа. Слои декодировщика последовательно передают результат следующему слою вместе с результатом кодировщика в качестве его входа.

Каждый кодировщик состоит из механизма самовнимания (вход из предыдущего слоя) и нейронной сети с прямой связью (вход из механизма самовнимания). Каждый декодировщик состоит из механизма самовнимания, механизма внимания к результатам кодирования (вход из механизма самовнимания и кодировщика) и нейронной сети с прямой связью.

В качестве примера можно привести такую языковую модель как BERT, основанную на архитекторе трансформер, предназначенную для предобучения языковых представлений с целью их последующего применения в широком спектре задач обработки естественного языка. BERT представляет собой нейронную сеть, основу которой составляет композиция кодировщиков трансформера. BERT является автокодировщиком. В каждом слое кодировщика применяется двустороннее внимание, что позволяет модели учитывать контекст с обеих сторон от рассматриваемого токена, а значит, точнее определять значения токенов. Пример устройства этапа предобучения модели на рисунке 1.4.

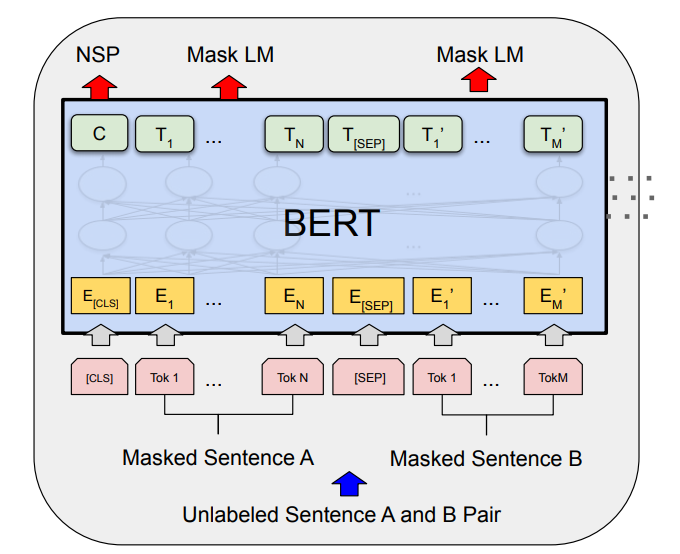


Рисунок 1.4 – Схема этапа предобучения BERT

Так же важно упомнить сети долгой краткосрочной памяти LSTM (Рисунок 1.5.) – тип рекуррентной нейронной сети, способный обучаться долгосрочным зависимостям. LSTM специально разработаны для устранения проблемы долгосрочной зависимости. Их специализация – запоминание информации в течение длительных периодов времени, поэтому их практически не нужно обучать.

Все рекуррентные нейронные сети имеют форму цепочки повторяющихся модулей нейронной сети. В стандартных РНС этот повторяющийся модуль имеет простую структуру, например, один слой tanh.

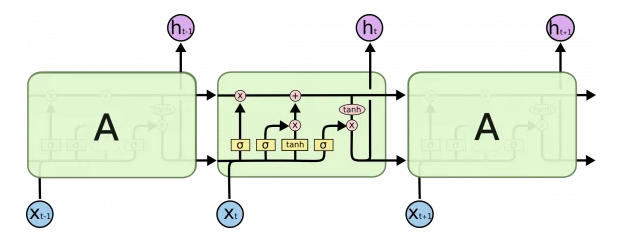


Рисунок 1.5 – Архитектура LSTM

* + 1. **Критерий эффективности функционирования разрабатываемого способа автоматизации определения деградаций в ответах HTTP-запросов**

Критерии эффективности разрабатываемого способа автоматизации определения аномалий в ответах HTTP-запросов представляют собой совокупность показателей, позволяющих оценить качество, надежность и практическую применимость предложенного решения. Для объективной оценки эффективности разработанного метода необходимо определить и описать ключевые критерии, которые будут использоваться в процессе тестирования и оценки модели. В данном разделе рассматриваются основные критерии, на основе которых будет оцениваться функционирование автоматизированной системы. Один из самых важных критериев – точность, представляет собой долю правильно классифицированных примеров по отношению к общему числу обработанных запросов, для системы банка, обученная модель должна иметь не менее 85% точности. Полнота измеряет способность модели выявлять все реальные случаи ошибок в ответах HTTP-запросов, следовательно, модель редко пропускает запросы с аномалией. F1-мера представляет собой среднее между точностью и полнотой, обеспечивая баланс между ними.

Таким образом, представим в виде таблицы основные параметры для сравнения существующих решений. Результат сравнения показан в таблице 1.1.

Таблица 1.1 – Сравнение существующих решений

|  |  |  |  |
| --- | --- | --- | --- |
| Критерии | ARIMA | Zabbix | Разработанный модуль |
| Тип данных | Временные ряды | Логи и метрики системы | Логи HTTP-запросов (текстовые данные) |
| Метод анализа | Математический прогноз на основе временных рядов | Правила и метрики с использованием пороговых значений | Классификация текстов с использованием BERT |
| Обработка текста | Нет | Частичная | Глубокая |
| Точность выявления деградаций | Средняя | Средняя | Высокая |
| Поддержка реального времени | Ограниченная | Реальная | Реальная |
| Гибкость настройки | Низкая | Высокая | Высокая |
| Сложность внедрения | Средняя | Низкая | Высокая |
| Возможность масштабирования | Ограниченная | Высокая | Высокая |
| Интеграция с другими системами | Ограниченная | Высокая | Высокая |
| Ключевое преимущество | Простота использования для временных рядов | Легкость настройки метрик | Высокая точность классификации текстовых данных |

Разработанное решение, основанное на Elastic и модели BERT, обладает преимуществами перед существующими подходами благодаря способности анализировать текстовые данные с высокой точностью.

* 1. **Модель угроз безопасности** 
     1. **Анализ информационной системы**

Elastic представляет собой комплексную информационную систему, основанную на поисковом движке Elasticsearch, которая широко используется для сбора, хранения, анализа и визуализации больших объёмов данных в реальном времени. В контексте автоматизации определения деградации в ответах HTTP-запросов, структура Elastic играет ключевую роль, обеспечивая эффективное управление логами и поддерживая высокую производительность аналитических процессов. Основными компонентами Elastic являются:

1. Elasticsearch: Elasticsearch (рисунок 1.6.) является ядром системы Elastic, представляя собой распределённый, RESTful поисковый и аналитический движок, способный обрабатывать большие объёмы данных с высокой скоростью. Он основан на библиотеке Apache Lucene и предоставляет мощные возможности для полнотекстового поиска и анализа данных. Архитектура Elasticsearch работает в комплексной архитектуре, состоящей из одного или нескольких узлов, объединённых в группу. Каждый узел может выполнять роль мастер-узла, узла данных или клиентского узла, обеспечивая масштабируемость и отказоустойчивость системы.

2. Logstash: Logstash служит для сбора, обработки и передачи данных в Elasticsearch. Он поддерживает множество входных плагинов для интеграции с различными источниками данных, включая файлы логов, базы данных, системы мониторинга и другие.

3. Пайплайны обработки: Logstash использует пайплайны, состоящие из входных, фильтрующих и выходных этапов, что позволяет гибко настраивать процессы обработки данных перед их индексированием в Elasticsearch.

4. Kibana: Kibana предоставляет пользовательский интерфейс для визуализации данных, хранящихся в Elasticsearch. С помощью Kibana можно создавать интерактивные панели управления, графики, диаграммы и другие визуальные представления данных, что облегчает анализ и мониторинг. Функциональность Kibana поддерживает различные типы визуализаций, фильтры, агрегации и возможности для создания пользовательских отчетов, что делает её незаменимым инструментом для аналитиков и инженеров по данным.

5. Beats: Beats представляют собой легковесные агенты для сбора данных с различных источников. Они устанавливаются на серверах и передают данные в Logstash или напрямую в Elasticsearch. Существует несколько типов Beats, таких как Filebeat (для сбора файлов логов), Metricbeat (для сбора метрик системы и приложений), Packetbeat (для анализа сетевого трафика) и другие, что позволяет адаптировать систему под конкретные задачи.

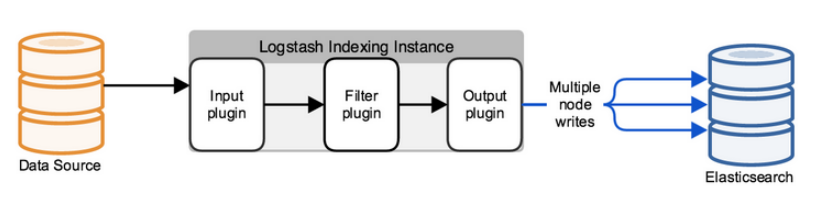


Рисунок 1.6 – Устройство Elasticsearch

Структуры данных в Elastic:

1. Индексы и документы: Индекс в Elasticsearch представляет собой логическое хранилище, содержащее документы с похожей структурой. Каждый документ — это единица информации, представляемая в формате JSON, содержащая поля с различными типами данных.

2. Маппинг (Mapping): Маппинг определяет схему индекса, описывая типы полей и их свойства. Правильное определение маппинга важно для оптимизации поиска и анализа данных.

3. Шардирование: Для обеспечения масштабируемости и высокой производительности, индекс делится на несколько частей, называемых шардами. Каждый шард является отдельным индексом и может храниться на разных узлах кластера.

4. Репликация: Для повышения отказоустойчивости, каждый шард может иметь реплики — копии, хранящиеся на других узлах. Это обеспечивает доступность данных даже при отказе отдельных узлов.

5. Кластеры: Кластер состоит из одного или нескольких узлов, работающих вместе для хранения данных и обработки запросов. В кластер могут входить разные типы узлов, такие как мастер-узлы (управляют кластером), узлы данных (хранят и обрабатывают данные) и клиентские узлы (обрабатывают запросы пользователей).

Применение Elastic для анализа логов HTTP-запросов:

1. Сбор и хранение логов: Filebeat может быть использован для сбора результатов работы сервера и передачи их в Logstash для дальнейшей обработки. Logstash выполняет предварительную обработку данных, включая фильтрацию, парсинг и структурирование данных перед индексированием в Elasticsearch.

2. Обработка и анализ данных: Elasticsearch обеспечивает быстрый доступ к данным и поддерживает сложные поисковые запросы, что позволяет эффективно искать и анализировать случаи деградации в ответах HTTP-запросов.

3. Агрегации и метрики: Elasticsearch поддерживает агрегации, которые позволяют вычислять статистические метрики, такие как среднее время ответа, частота ошибок и другие показатели производительности.

4. Визуализация и мониторинг: Kibana используется для создания панелей управления, отображающих ключевые метрики и тенденции, связанные с деградацией HTTP-ответов. Это позволяет оперативно выявлять аномалии и принимать меры для их устранения.

5. Настройка уведомлений: В Kibana можно настроить уведомаления, которые будут оповещать администраторов о превышении пороговых значений метрик деградации, обеспечивая своевременное реагирование на проблемы.

Преимущества использования Elastic в контексте автоматизации определения деградации

1. Масштабируемость и производительность: Elastic позволяет обрабатывать большие объёмы данных в реальном времени благодаря распределённой архитектуре и эффективному шардингу.

2. Гибкость и адаптивность: Система легко адаптируется под различные форматы данных и требования аналитики, что делает её подходящей для различных сценариев мониторинга сервисов.

3. Интеграция с инструментами машинного обучения: Elastic интегрируется с инструментами машинного обучения, что позволяет автоматически обнаруживать аномалии и прогнозировать потенциальные проблемы на основе исторических данных.

4. Удобство визуализации: Kibana предоставляет мощные средства для визуализации данных, что облегчает анализ и принятие решений на основе полученных метрик.

Анализ структуры информационной системы Elastic показывает, что она предоставляет все необходимые инструменты и возможности для эффективного сбора, хранения, анализа и визуализации ответов HTTP-запросов. Распределённая архитектура, поддержка масштабируемости, интеграция с методами машинного обучения и удобные средства визуализации делают Elastic оптимальным выбором для автоматизации определения деградации в ответах HTTP-запросов. Это позволяет не только повысить надёжность и производительность веб-сервисов, но и обеспечить своевременное обнаружение и устранение возникающих проблем, что является ключевым фактором успешной деятельности современных информационных систем.

Определим класс защищённости исследуемой системы в соответствии с Руководящим документом от 30 марта 1992 г. «Средства вычислительной техники. Защита от несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации». В таблице \_ приведено сравнение показателей защищенности системы с требуемыми показателями защищенности для класса защищенности 6.

Таблица 1.2. Сравнение показателей защищенности системы с требуемыми показателями защищенности для класса защищенности 6

|  |  |  |
| --- | --- | --- |
| Наименование показателя | Система, соответствующая Классу Защищенности 6 | Текущие показатели исследуемой системы |
| Дискреционный принцип контроля доступа | + | + |
| Идентификация и аутентификация | + | + |
| Регистрация | + | + |
| Руководство для пользователя | + | + |
| Руководство по КСЗ | + | + |
| Тестовая документация | + | + |
| Конструкторская (проектная) документация | + | - |

Таким образом, исследуемая система Elastic соответствует требованием класса 6 по обеспечению базовых механизмов идентификации и аутентификации пользователей, а также поддерживает базовое руководство для пользователя, так же система частично соответствует классу 5.

* + 1. **Виды рисков (угроз) и типовые негативные последствия от реализации угроз безопасности и информации**

Проведем анализ рисков и типовых негативных последствий в рамках методики ФСТЭК, чтобы выявить ключевые угрозы безопасности для данных при их обработке с помощью Elastic.

В таблице 1.3. приведены возможные виды рисков и негативные последствия в случае успешной атаки на информационную систему.

Таблица 1.3 — Виды рисков и негативных последствий

|  |  |  |
| --- | --- | --- |
|  | Виды рисков (ущерба) | Возможные типовые негативные  последствия |
| У1 | Ущерб физическому лицу | Нарушение неприкосновенности частной жизни.  Нарушение тайны переписки, телефонных переговоров, иных сообщений. Финансовый и материальный ущерб. Нарушение конфиденциальности (утечка) персональных данных.  Разглашение персональных данных граждан. |
| У2 | Риски юридическому лицу, индивидуальному предпринимателю, связанные с хозяйственной деятельностью | Нарушение законодательства Российской Федерации.  Недополучение ожидаемой (прогнозируемой) прибыли.  Ущерб деловой репутации. Финансовые санкции и штрафы за нарушение требований законодательства в области защиты данных.  Потеря конкурентных преимуществ. |
| У3 | Риски экономического ущерба | Увеличение операционных затрат на восстановление данных и систем.  Прямые финансовые потери от недоступности услуг.  Потеря клиентов и снижение доходов. Дополнительные затраты на устранение последствий инцидента.  Потеря инвесторов и снижение рыночной стоимости. |
| У4 | Риски для непрерывности бизнеса | Нарушение работы критически важных бизнес-процессов.  Простой или полная остановка системы. Снижение производительности.  Увеличение времени восстановления работоспособности системы.  Проблемы с поставщиками и партнёрами из-за задержек в бизнес-процессах. |
| У5 | Риски утраты конфиденциальной информации | Утечка коммерческой тайны и внутренней информации.  Разглашение данных о клиентах и партнёрах.  Снижение доверия клиентов.  Повышенные риски целенаправленных атак (фишинг, социальная инженерия) из-за утечки данных.  Необходимость дополнительных мер безопасности. |
| У6 | Репутационные риски | Потеря доверия со стороны клиентов и партнеров.  Негативные публикации в СМИ и социальных сетях.  Ухудшение имиджа компании.  Отток клиентов и снижение числа новых пользователей.  Увеличение затрат на маркетинг и восстановление репутации. |
| У7 | Юридические и правовые риски | Нарушение требований законодательства (например, GDPR, 152-ФЗ).  Вероятность судебных исков со стороны клиентов или партнеров.  Штрафы и санкции от регулирующих органов.  Ограничение доступа к определенным рынкам.  Увеличение затрат на юридическую защиту. |

* + 1. **Модель нарушителя**

Модель нарушителя может включать как внутренние, так и внешние угрозы. Внутренние угрозы чаще связаны с инсайдерами — сотрудниками или подрядчиками, имеющими доступ к системе, а внешние — с киберпреступниками, которые могут попытаться атаковать систему через сетевые уязвимости.

Таблица 1.4 — Возможные цели реализации угроз безопасности информации нарушителями

|  |  |  |  |
| --- | --- | --- | --- |
|  | Виды нарушителя | Категории нарушителя | Возможные цели реализации угроз безопасности информации |
| 1 | Разработчики программных,  программно-аппаратных средств | Внутренний | Получение финансовой или иной материальной выгоды.  Непреднамеренные, неосторожные действия, связанные с ошибками в коде, которые могут привести к уязвимостям. |
| 2 | Лица, привлекаемые для установки, настройки, испытаний, пусконаладочных и иных видов работ, в т ч подрядчики | Внутренний | Получение материальной выгоды, доступ к конфиденциальной информации. Ненамеренные действия, ведущие к ошибкам конфигурации, снижению производительности и деградациям в системе. |
| 3 | Авторизованные пользователи систем и сетей | Внутренний | Любопытство, желание самоутверждения или месть за ранее совершенные действия со стороны руководства. Возможен несанкционированный доступ к конфиденциальным данным и модификация логов. |
| 4 | Администраторы системы и базы данных | Внутренний | Получение доступа к данным с целью их кражи или модификации. Злоупотребление привилегиями для получения выгоды. Возможность манипуляции настройками системы для сокрытия следов действий. |
| 5 | Внешние хакеры и киберпреступники | Внутренний | Эксплуатация уязвимостей системы Elastic для получения доступа к конфиденциальным данным. Снижение производительности и доступности системы, организация DDoS-атак или внедрение вредоносного ПО. |
| 6 | Конкуренты | Внутренний | Получение коммерческой информации для усиления конкурентных позиций. Возможные действия включают кражу данных о клиентской базе или попытки саботажа через снижение производительности системы. |
| 7 | Специалисты по социальной инженерии | Внутренний | Получение доступа к учетным данным через обман сотрудников (фишинг, подделка запросов). Использование учетных данных для кражи данных или модификации системы в целях дестабилизации работы. |

Теперь рассмотрим уровни возможностей нарушителей по реализации угроз безопасности информации. Результаты представлены в таблице 1.5.

Таблица 1.5 — Уровни возможностей нарушителей по реализации угроз безопасности информации

|  |  |  |  |
| --- | --- | --- | --- |
|  | Уровень возможностей  нарушителей | Возможности нарушителей по реализации угроз безопасности информации | Виды нарушителей |
| Н1 | Нарушитель, обладающий базовыми возможностями | Нарушители с базовыми возможностями могут использовать известные (документированные) уязвимости и общедоступные инструменты для реализации угроз. Их действия направлены на компрометацию стандартных механизмов защиты. | Авторизованные пользователи системы Elastic и сетей, обладающие базовым доступом и ограниченными привилегиями. |
| Н2 | Нарушитель, обладающий базовыми повышенными возможностями | Нарушители с повышенными возможностями могут эксплуатировать как известные, так и неизвестные (недокументированные) уязвимости, используя специально разработанные инструменты для обхода стандартных средств защиты. | Лица, привлекаемые для настройки и тестирования системы, администраторы системы Elastic и базы данных, обладающие более высоким уровнем привилегий и доступом к настройкам системы. |
| Н3 | Нарушитель, обладающий средними возможностям | Нарушители с профессиональными возможностями способны разрабатывать сложные, целевые атаки на систему, используя неизвестные уязвимости, многоступенчатые подходы и скрытые методы для компрометации безопасности. | Внешние хакеры и киберпреступники, имеющие доступ к продвинутым инструментам и ресурсам для проведения атак; конкуренты, заинтересованные в получении конфиденциальной информации для получения преимуществ. |

* + 1. **Модель угроз**

Система Elastic, используемая для мониторинга и анализа логов, подвергается различным угрозам, которые могут повлиять на её работоспособность, безопасность данных и доступность функционала. Основные векторы атак, представляющие угрозы для данной системы, включают:

* Несанкционированный доступ: Попытки получить доступ к системе мониторинга или логам без соответствующих привилегий. Нарушители могут пытаться получить доступ к конфиденциальным данным, используя слабые точки в системе аутентификации или обходя защитные механизмы. Подобные действия могут привести к раскрытию информации о деградациях и других важных данных, касающихся инфраструктуры.
* Атаки на аутентификацию: Включают в себя перебор паролей, фишинговые атаки, взлом учётных данных и другие методы нарушения механизмов аутентификации. Успешная атака на аутентификацию может позволить злоумышленнику получить несанкционированный доступ к системе, манипулировать логами или скрывать собственные действия, что усложнит идентификацию реальных деградаций.
* Недостатки в авторизации и разграничении доступа: Неправильно настроенные права доступа могут позволить злоумышленникам или неквалифицированным сотрудникам получать доступ к данным или функционалу, к которым они не должны иметь доступа. Например, доступ к административным функциям, модификация конфигураций или манипуляции с логами могут привести к искажению данных о деградациях и ложным срабатываниям системы мониторинга.
* SQL Injection и Command Injection: Атаки, направленные на внедрение вредоносных SQL-кодов или команд в запросы к базе данных. Такие уязвимости могут позволить злоумышленникам получить несанкционированный доступ к данным, модифицировать или удалять критически важные данные о логах и деградациях в системе Elastic.
* Деградация сервисов (Denial-of-Service, DoS): Попытки вызвать перегрузку системы Elastic для временной или постоянной остановки её функционирования. В случае успешной DoS-атаки система может стать недоступной для обработки и анализа логов, что повлияет на своевременное выявление деградаций и оперативное реагирование на них.
* Эксплуатация уязвимостей в API: Нарушители могут использовать уязвимости в открытых API-интерфейсах Elastic для получения несанкционированного доступа или выполнения вредоносных операций. Это может включать выполнение несанкционированных команд, манипуляцию данными логов или изменение параметров системы, что потенциально может скрыть настоящие деградации или вызвать ложные срабатывания.
  1. **Типовые негативные последствия от многочисленных деградаций**

**1.4.1 Оценка экономических потерь от многочисленных деградаций**

Как говорилось раннее модель будет внедрена в банковскую ИС, что требует обеспечения высокой степени надежности и доступности сервисов, поскольку любая деградация в ответах HTTP-запросов может привести к значительным экономическим потерям. В условиях интенсивного использования цифровых каналов обслуживания клиентов, банковские учреждения сталкиваются с рядом финансовых и нефинансовых рисков, связанных с ухудшением качества предоставляемых услуг. Одной из основных экономических потерь, связанных с деградацией HTTP-ответов, является снижение производительности приложений, что напрямую влияет на способность банка обрабатывать транзакции и обслуживать клиентов. Например, задержки в обработке платежей или запросов на баланс могут привести к следующим финансовым последствиям:

1. Увеличение операционных задач: Простой системы требует дополнительных ресурсов для её восстановления и устранения возникших проблем. Это включает затраты на привлечение IT-специалистов, использование резервных систем и проведение нештатных мероприятий по восстановлению работоспособности.
2. Снижение доходов: Недоступность ключевых сервисов, таких как онлайн-банкинг или мобильные приложения, может привести к потере транзакций, снижению объёмов операций и, как следствие, уменьшению доходов банка. Например, если система обработки кредитных заявок работает с перебоями, это может замедлить процесс выдачи кредитов и снизить общий объём выдаваемых кредитов.
3. Потеря доверия и репутации: Постоянные сбои и низкое качество обслуживания могут привести к оттоку клиентов в конкурирующие учреждения, предлагающие более стабильные и надёжные сервисы. Согласно исследованиям, более 60% клиентов готовы сменить банк из-за неудовлетворительного качества обслуживания.
4. Юридические и нормативные последствия: Клиенты, пострадавшие от деградации сервисов, могут предъявить иски к банку за причинённые убытки, что дополнительно увеличивает финансовые потери и негативно сказывается на репутации учреждения.

Для наглядности можно представить график, показывающий корреляцию между уровнем деградации сервисов и экономическими потерями банка:

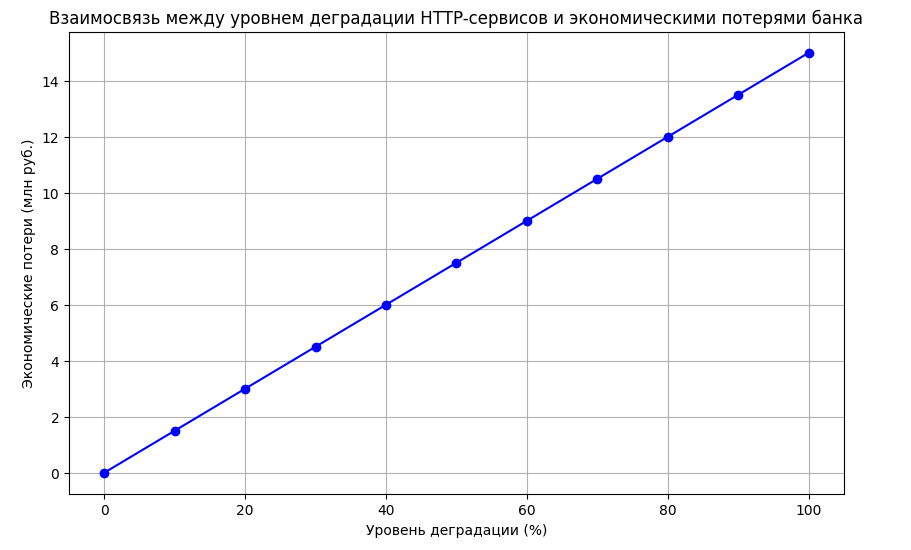


Рисунок 1.7 — Иллюстрация экономических потерь

Оценка экономических потерь от деградации HTTP-ответов подчёркивает критическую важность автоматизации процесса их обнаружения и устранения. Внедрение искусственного интеллекта для мониторинга и анализа логов позволяет значительно снизить риски, связанные с эксплуатационными сбоями, повысить уровень обслуживания клиентов и обеспечить соответствие нормативным требованиям. Таким образом, разработка эффективных методов автоматизации определения деградаций в ответах HTTP-запросов является не только технической необходимостью, но и стратегическим решением для обеспечения устойчивости и конкурентоспособности банковских информационных систем.

**1.4.2. Оценка потенциальной прибыли при автоматизации определения деградаций ответах HTTP-запросов**

Автоматизация процесса определения деградаций в ответах HTTP-запросов с использованием искусственного интеллекта предоставляет значительные экономические преимущества для банковской информационной системы. Внедрение таких решений способствует не только снижению операционных затрат, но и увеличению доходов, улучшению качества обслуживания клиентов и укреплению конкурентных позиций банка на рынке.

* 1. **Вывод о целесообразности разработки в качестве способа повышения эффективности мониторинга и обнаружения деградаций в ответах HTTP-запросов данного программного продукта и уточненная постановка основной задачи**

Из анализа существующих решений в области мониторинга веб-сервисов и обнаружения деградаций в ответах HTTP-запросов видно, что большинство доступных инструментов основаны на традиционных методах анализа логов и статистических подходах. Эти методы часто не способны эффективно справляться с большими объёмами данных и сложными паттернами аномалий, что ограничивает их применимость в современных условиях высокой динамики и нагрузки на веб-сервисы. Кроме того, многие коммерческие решения имеют закрытые алгоритмы и ограниченные возможности по настройке под специфические требования конкретных информационных систем.

На основании вышеизложенного наиболее рациональным вариантом решения задачи автоматизации определения деградаций в ответах HTTP-запросов является разработка собственного программного продукта. Такой подход позволит создать гибкую и адаптивную систему, способную учитывать особенности конкретной банковской информационной системы Elastic и обеспечивать высокую точность обнаружения проблемных запросов.

Таким образом, в качестве способа повышения эффективности мониторинга и обнаружения деградаций в ответах HTTP-запросов в банковской ИС предлагается разработать программный продукт с следующими функциями:

1. Сбор и сохранение логов HTTP-запросов:

- Автоматическое сохранение копий всех входящих и исходящих HTTP-запросов и ответов.

2. Реальное время мониторинга:

- Получение оповещений о новых HTTP-запросах и автоматический анализ их ответов на предмет деградаций.

3. Анализ содержания ответов:

- Определение тематики и содержания HTTP-ответов с использованием методов обработки естественного языка (NLP).

4. Сопоставление с правами доступа:

- Соотнесение содержания ответов с правами доступа пользователей к соответствующей информации для выявления несанкционированного доступа.

5. Категоризация запросов:

- Классификация адресатов запросов по категориям (например, по типу обрабатываемой информации) для более точного анализа.

6. Выявление аномалий и подозрительных действий:

- Определение фактов передачи конфиденциальной информации и классификация запросов как разрешённых или подозрительных на основе обученной модели искусственного интеллекта.

7. Отчётность и визуализация:

- Составление подробных отчётов по результатам проверки HTTP-запросов, включая визуализацию данных для облегчения анализа и принятия решений.

При разработке программного продукта необходимо учитывать следующие критерии:

- Точность обнаружения: Стремление к достижению уровня точности не ниже 80% для минимизации ложных срабатываний и пропусков.

- Применимость: Продукт должен быть адаптирован под специфические требования и архитектуру банковской ИС Elastic.

- Простота использования: Интерфейс и функциональность системы должны быть интуитивно понятными для пользователей с разным уровнем технической подготовки.

- Незаметность: Система должна работать в фоновом режиме без значительного влияния на производительность информационной системы.

По завершении разработки программного продукта требуется провести оценку экономических затрат на его создание и эксплуатацию. Это включает в себя расходы на разработку программного обеспечения, приобретение необходимых вычислительных ресурсов, обучение персонала и техническую поддержку системы. Ожидается, что внедрение автоматизированной системы обнаружения деградаций позволит существенно снизить операционные издержки, связанные с ручным анализом логов, повысить уровень безопасности и надежности информационной системы, а также снизить риски финансовых потерь и ухудшения репутации банка вследствие невыявленных деградаций.

Уточненная постановка основной задачи: Разработать и внедрить программный продукт на основе искусственного интеллекта, способный автоматически определять деградации в ответах HTTP-запросов в информационной системе Elastic банка, обеспечивая высокую точность обнаружения, адаптивность к специфическим требованиям системы и экономическую эффективность эксплуатации.

1. **МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ФУНКЦИОНИРОВАНИЯ СИСТЕМЫ**
   1. **Постановка задачи**

Автоматизация определения деградации в ответах HTTP-запросов представляет собой задачу классификации, где необходимо определить, является ли ответ на HTTP-запрос нормальным или содержит признаки деградации. Для формализации этой задачи введем следующие обозначения и определения.

* + 1. **Обозначение и определения**

Пространство входных данных: множество всех возможных HTTP – ответов, представленных в виде текстовых строк или структурированных данных.

Пространство выходных данных: , где соответствует нормальному ответу, соответствует ответу с деградацией.

Обучающая выборка: набор размеченных данных

Модель классификации:

Функция ответа x предсказывает метку класса

* + 1. **Цель**

Целью является построение такой модели , которая минимизирует вероятность ошибочной классификации на новых, ранее не встречавшихся данных. Это достигается путем обучения модели на обучающей выборке и последующей валидации на тестовой выборке.

* 1. **Преобразование данных**

Для применения методов машинного обучения и нейронных сетей необходимо преобразовать текстовые данные в числовые представления.

* + 1. **Токенизация**

Токенизация — процесс разбиения текстовых данных на отдельные токены (слова, символы или подслова). Пусть текстовый HTTP – ответ. Токенизация: , где токен, длина последовательности токенов.

* + 1. **Векторизация**

Каждому токену сопоставляется числовой вектор из эмбеддингового пространства. Используется эмбеддинг-функция последовательность векторов:

* 1. **Математическое описание модели нейронной сети**

Для решения задачи классификации применяется нейронная сеть на основе архитектуры трансформера, например, BERT (Bidirectional Encoder Representations from Transformers).

* + 1. **Архитектура модели BERT**

Входные данные: Последовательность эмбеддингов

Трансформер: Использует механизмы внимания для обработки последовательности.

* Механизм самовнимания (Self-Attention)
  + Ключи, запросы и значения:
    - Запрос:
    - Ключ:
    - Значение:

и

* Вычисление внимания:
  + Коэффициенты внимания между токенами
  + Выходной вектор для токена :
* Многоголовое внимание (Multi-Head Attention)
  + Параллельное выполнение нескольких механизмов внимания для захвата различных типов зависимостей.
  + Выходы объединяются и проходят через линейный слой.
    1. **Добавление позиционной информации**

Поскольку трансформеры не учитывают порядок последовательности, вводятся позиционные эмбеддинги , которые добавляются к входным эмбеддингам:

* + 1. **Выход модели и классификация**
* Получение скрытого представления:
  + Используется специальный токен [CLS] в начале последовательности, чье скрытое представление используется для классификации.
* Классификационный слой:
  + Применяется полносвязный слой с функцией активации softmax или sigmoid для получения вероятности принадлежности к классу деградации:
    - Для бинарной классификации с использованием сигмоиды: , где матрица весов входного слоя, b – смещение, сигмоидная функция:
    1. **Функция потерь**

Для задачи бинарной классификации используется бинарная кросс-энтропия: , где y – истинная метка класса, предсказанная моделью вероятность принадлежности к классу деградации.

* + 1. **Обучение модели**
* Оптимизатор: Используется алгоритм AdamW, модификация Adam с коррекцией весового распада потерь по отношению к весам: где параметры модели, скорость обучения, градиент функции потерь.
* Регуляризация:
  + Dropout: Случайное отключение нейронов в процессе обучения для предотвращения переобучения.
  + Весовой распад: Добавление штрафа за большие значения весов
  1. **Метрики оценки качества модели**

Для оценки эффективности модели мы используем следующие метрики.

* + 1. **Матрица ошибок**

Таблица 2.1 — Матрица ошибок

|  |  |  |
| --- | --- | --- |
|  | Предсказано положительное | Предсказано отрицательное |
| Положительное (y=1) | Истинно положительное (TP) | Ложно отрицательное (FN) |
| Отрицательное (y=0) | Ложно положительное (FP) | Истинно отрицательное (TN) |

* + 1. **Точность (Accuracy)**
    2. **Точность (Precision)**
    3. **Полнота (Recall)**
    4. **F1-мера (F1 Score)**
    5. **ROC-кривая и AUC**
* ROC-кривая (Receiver Operating Characteristic) отображает зависимость между долей истинно положительных и ложно положительных результатов при изменении порога классификации.
* AUC (Area Under Curve) — площадь под ROC-кривой, измеряет способность модели различать классы.
  1. **Преобразование задачи в математическую оптимизацию**

Задача обучения модели сводится к минимизации функции потерь на обучающей выборке:

Где параметры модели, N – размер обучающей выборки.

* 1. **Математическое описание процесса обнаружения деградации**

После обучения модель используется для предсказания на новых данных.

* Вход: Новый HTTP – ответ
* Предсказание:
* Решение:
  + Если заданный порог, то считаем, что обнаружена деградация.
  + Иначе считаем ответ нормальным.
  1. **Выводы**

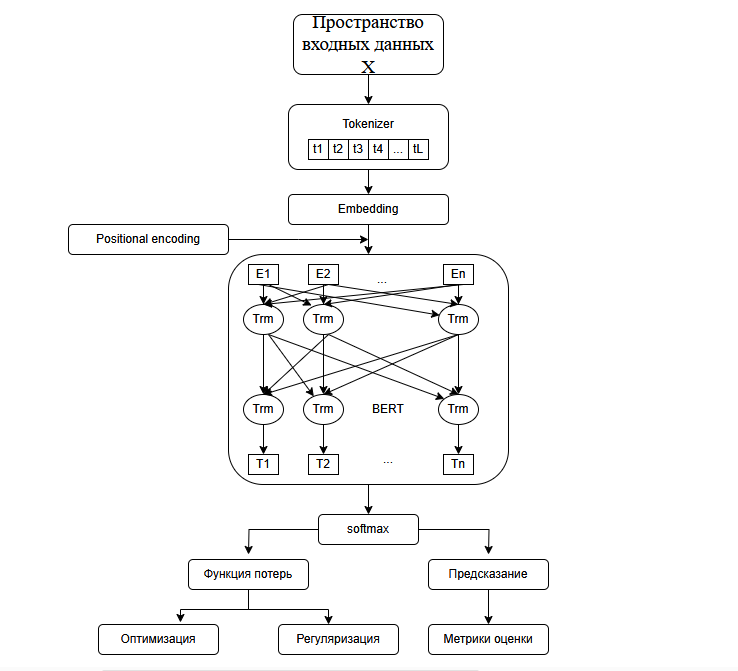
Представленная математическая модель формализует задачу автоматизации определения деградации в ответах HTTP-запросов как задачу бинарной классификации. Использование модели нейронной сети на основе архитектуры трансформера, такой как BERT, позволяет эффективно обрабатывать текстовые данные и выявлять сложные зависимости, характерные для деградированных ответов. Модель обучается путем минимизации функции потерь на обучающей выборке, а ее эффективность оценивается с помощью стандартных метрик классификации. Структуру модели можно представить в виде схемы, как показано на рисунке 2.1.  


Рисунок 2.1 – Схема модели

Таблица 2.2 — Обозначения, используемые в математической модели

|  |  |
| --- | --- |
| **Обозначение** | **Описание** |
|  | Пространство входных данных (HTTP-ответы) |
|  | Пространство выходных данных |
|  | Размер обучающей выборки |
|  | i-й HTTP-ответ |
|  | Метка класса для |
|  | Модель классификации |
|  | Параметры модели |
|  | Функция потерь |
|  | Предсказанная вероятность деградации |
|  | Порог классификации |
|  | Эмбеддинг i-го токена |
|  | Скрытое представление токена [CLS] |
|  | Сигмоидная функция |
|  | Истинно положительные |
|  | Истинно отрицательные |
|  | Ложно положительные |
|  | Ложно отрицательные |

Важно отметить, что модель может быть расширена для многоклассовой классификации, если требуется различать несколько типов деградаций, так же при обучении модели важно учитывать баланс классов в обучающей выборке. При сильном дисбалансе рекомендуется применять методы балансировки.

1. **СПОСОБ АВТОМАТИЗАЦИИ ОПРЕДЕЛЕНИЯ ДЕГРАДАЦИИ В ОТВЕТАХ HTTP-ЗАПРОСОВ В ВИДЕ ПРОГРАММНОГО МОДУЛЯ**
   1. **Алгоритм работы программного модуля**

Рассмотрим алгоритм работы программного модуля, представляющего собой последовательность шагов, обеспечивающих автоматизированное обнаружение деградаций в ответах HTTP – запросов. Алгоритм работы состоит из двух основных фаз:

* + 1. **Сбор и предобработка данных**

Для обучения модели под требуемы условия определения деградации, собираются логи HTTP-запросов из информационной системы Elastic. В данные входят текстовые сообщения, метки, указывающие на корректные или деградированные статусы запросов. Так же важно провести предобработку данных, очистить текст от лишних символов, нормализацию, удаление стоп-слов и токенизацию.

Изначально, вручную, с помощью фильтров в Elastic была определена выборка фраз, обозначающих деградированный ответ, а также выборка удовлетворяющих ответов. После чего с помощью кода написанного на Groovy, с использованием туннеля к Elastic системы находящейся на «бою», были получены логи в csv файл, каждая запись помечалась меткой: 0 для нормальных запросов и 1 для деградированных. В дальнейшем из логов вручную была удалена лишняя информация, не относящаяся к задаче анализа.

Для токенизации используется токенизатор, соответствующий модели BERT, который разбивает текст на токены в соответствии с внутренней структурой модели. На рисунке 3.1. представлен пример разбиения предложения на токены.

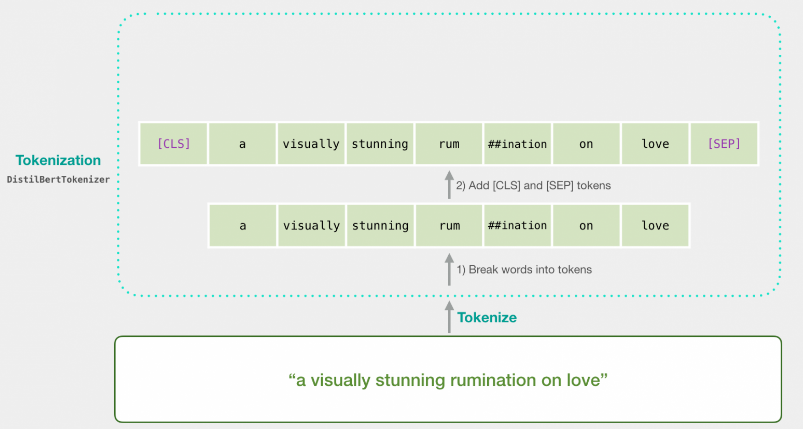


Рисунок 3.1 — Разбиение на токены

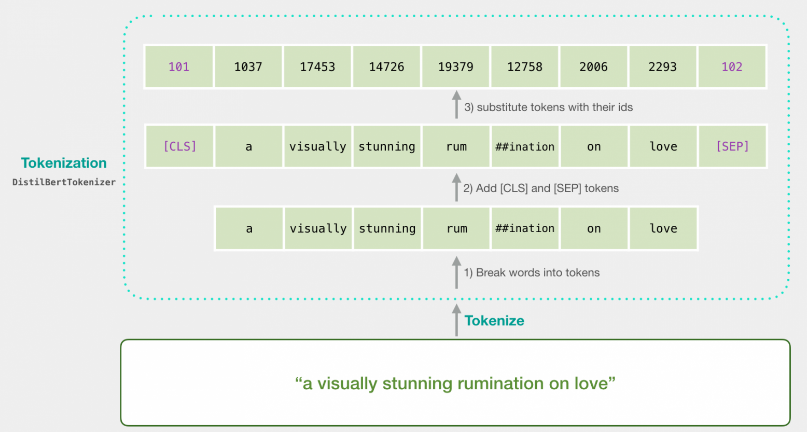
Далее токены преобразуются в числовые идентификаторы, которые могут быть использованы в нейронной сети, как представлено на рисунке 3.2.   


Рисунок 3.2 — Преобразование в числовые идентификаторы

На рисунке 3.3. показан участок кода, который отвечает за работу с токенами

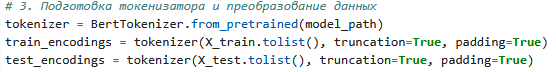


Рисунок 3.3 — Взаимодействие с токенами

Далее данные разделяются на обучающую и тренировочную выборку в определенной пропорции, в моем случае это 70% и 30%. Используются специальные классы для создания объектов датасета, совместимых с фреймворком обучения



Рисунок 3.4 — Выборка данных

В коде данная процедура выглядит так, как указано на рисунке 3.5.  

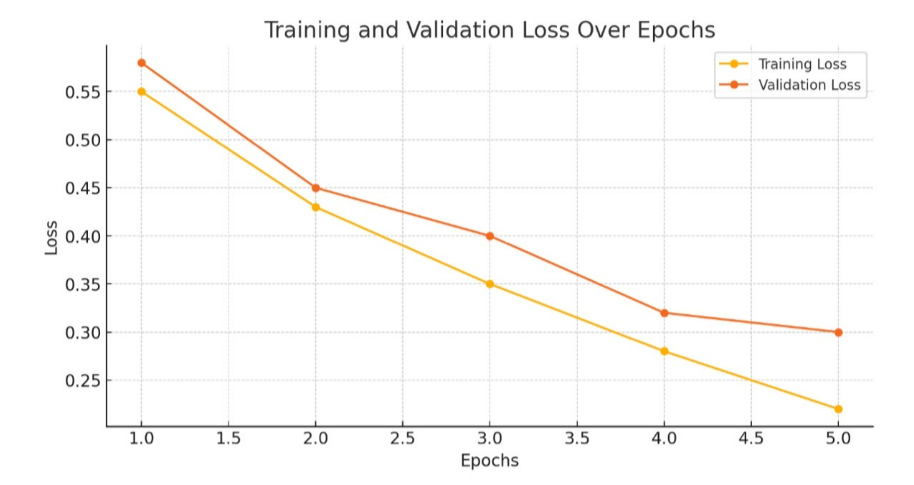

Рисунок 3.5 — Разделение данных на выборку

* + 1. **Настройка и инициализация модели**

Для успешного решения задачи автоматизации определения деградаций в ответах HTTP-запросов необходимо тщательно настроить и инициализировать модель искусственного интеллекта. Использование предобученной модели BERT (Bidirectional Encoder Representations from Transformers) обосновано её высокой эффективностью в обработке естественного языка и способности улавливать контекстные зависимости в тексте. Это особенно важно при анализе логов HTTP-запросов, где значение сообщений может сильно зависеть от контекста.

Использование библиотеки Hugging Face Transformers предоставляет удобные инструменты для работы с предобученными моделями и их адаптации к конкретным задачам.

На этапе предобучения модели определения деградаций на основе BERT, производится на раннее составленном датасете из реальных логов, обучение производится на 20 эпохах с ранней остановкой в случае, если модель не улучшила метрики на протяжении 3 эпох.



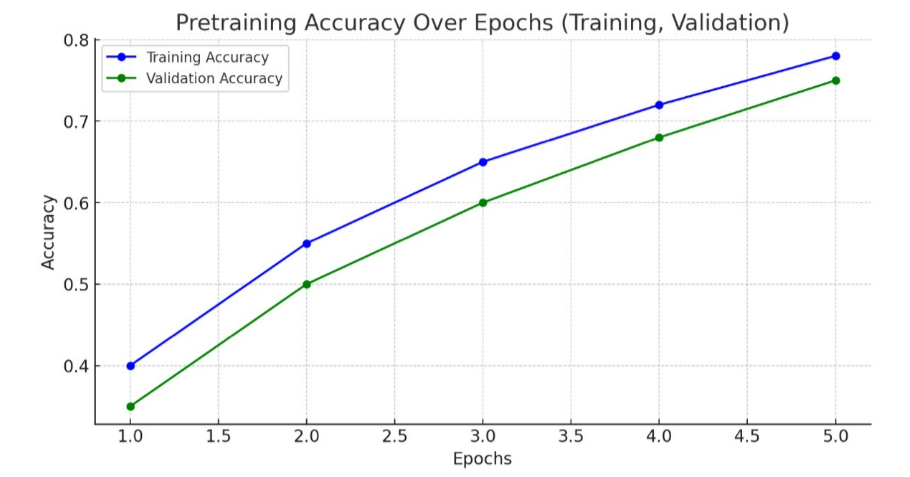


Рисунок 3.6 – Графики изменения точности и функции потерь на этапе предобучения

Предобученная модель BERT обучена на задаче маскированного моделирования языка и прогнозирования следующего предложения. Для использования её в задаче бинарной классификации необходимо внести некоторые изменения, например, определение количества классов, поскольку задача состоит в классификации логов на нормальные и деградированные, устанавливаем num\_labels=2. Настройка выходного слоя отвечает за добавление или перенастройка классификационного слоя (полносвязного слоя) для соответствия количеству классов.

Для эффективного обучения модели необходимо определить оптимизатор и функцию потерь. В качестве оптимизатора используется AdamW, который хорошо зарекомендовал себя при обучении трансформерных моделей.

Параметры оптимизатора: lr (learning rate): скорость обучения, обычно устанавливается на уровне 2e-5. eps (epsilon): параметр для численной стабильности, часто используется значение 1e-8.

Для задачи классификации используется функция CrossEntropyLoss, которая встроена в модель BertForSequenceClassification.

Оптимальные значения гиперпараметров обеспечивают баланс между скоростью обучения и точностью модели:

Количество эпох (epochs): В нашем случае устанавливаем epochs = 20.

Размер батча (batch\_size): Зависит от доступной памяти GPU/CPU. В нашем случае используется значение batch\_size = 16.

Планировщик скорости обучения помогает динамически изменять learning rate в процессе обучения: Используется get\_linear\_schedule\_with\_warmup из transformers.optimization. Позволяет установить фазу разогрева (warmup) и линейное уменьшение скорости обучения.

В дальнейшем проходит этап дообучения модели и на графиках можно заметить явные улучшения точности и уменьшение потерь, в сравнении с этапом предобучения, результат можно увидеть на рисунке 3.7.



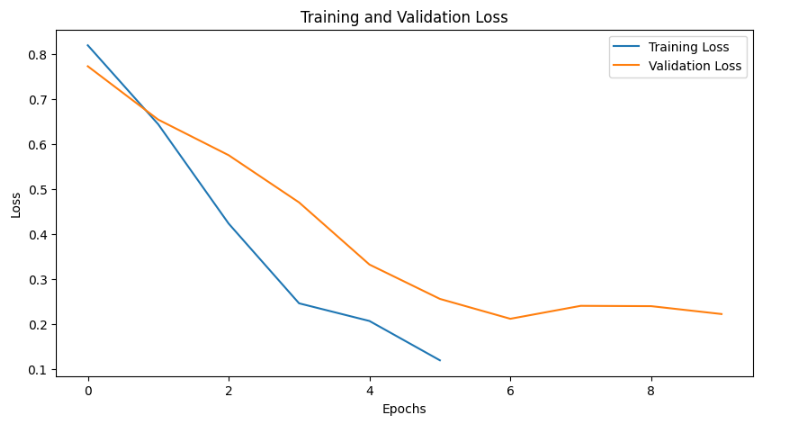


Рисунок 3.7 – Графики изменения точности и функции потерь на этапе дообучения

Настройка и инициализация модели являются критически важными этапами в разработке эффективного решения для автоматизации определения деградаций в ответах HTTP-запросов. Правильный выбор предобученной модели, корректная адаптация её к специфике задачи, оптимальная настройка гиперпараметров и тщательная подготовка данных обеспечивают высокую точность и надёжность модели. В результате разработанная модель способна эффективно классифицировать логи HTTP-запросов, способствуя своевременному обнаружению и устранению деградаций в банковской информационной системе.

* 1. **Инсталляция и настройка разработанного программного модуля**

Для корректной работы разработанного программного модуля для автоматизации определения деградации в ответах HTTP-запросов, было необходимо получить необходимые доступы с корпоративного устройства и установить следующие компоненты: Python 3.10, Torch, Transformers, pandas, scikit-learn, Gradle-7.3.2, Java OpenJDK-17-jdk

Далее тут будет описан процесс внедрения модели в ИС

* 1. **Обоснование выбора модели**

Выбор подходящей модели машинного обучения является критически важным этапом в разработке системы автоматизации определения деградации в ответах HTTP-запросов. Правильный выбор модели обеспечивает высокую точность, эффективность и надежность решения, соответствующего специфике задачи и требованиям банковской информационной системы. В данном разделе подробно рассматриваются критерии выбора модели, анализируются альтернативные подходы и обосновывается использование модели BERT для решения поставленной задачи.

При выборе модели машинного обучения были учтены следующие ключевые критерии:

**3.3.1. Способность обрабатывать текстовые данные**

Поскольку логи HTTP-запросов представляют собой текстовые данные, модель должна эффективно работать с естественным языком и улавливать семантические и синтаксические особенности текста.

**3.3.2. Высокая точность и надежность**

Модель должна обеспечивать высокую точность классификации, минимизируя количество ложноположительных и ложноотрицательных срабатываний, что особенно важно в банковской сфере.

**3.3.3. Способность учитывать контекст**

Деградация в ответах HTTP-запросов может проявляться в различных формах и зависеть от контекста. Модель должна быть способна учитывать контекстные зависимости в данных.

**3.3.4. Производительность и масштабируемость**

Модель должна быть способна обрабатывать большие объемы данных в реальном времени без существенного снижения производительности.

**3.3.5. Простота интеграции и использования**

Модель должна быть относительно простой в реализации и интеграции в существующую инфраструктуру, с возможностью дальнейшего развития и поддержки.

При рассмотрении различных моделей и методов были проанализированы следующие варианты:

Традиционные методы машинного обучения: логистическая регрессия, Метод опорных векторов (SVM), Деревья решений и случайные леса.

Преимущества: простота реализации и интерпретации, низкие вычислительные затраты.

Недостатки: ограниченная способность обрабатывать неструктурированные текстовые данные без сложной предобработки, неэффективность в улавливании сложных и нелинейных зависимостей в данных.

Требуется значительное усилие для создания и настройки признакового пространства (feature engineering).

**3.3.6. Классические нейронные сети**

- Многослойные персептроны (MLP)

- Сверточные нейронные сети (CNN) для текстовых данных

Преимущества: способность моделировать нелинейные зависимости, более высокая производительность по сравнению с традиционными методами.

Недостатки: ограниченная способность учитывать длинные последовательности и контекст, требуется большая обучающая выборка для достижения высокой точности.

**3.3.7. Рекуррентные нейронные сети (RNN)**

- Локально связные рекуррентные сети

- Долгая краткосрочная память (LSTM)

- Управляемые рекуррентные единицы (GRU)

Преимущества: хорошо справляются с последовательными данными и могут учитывать временные зависимости, способны моделировать контекст в определенной степени.

Недостатки: проблемы с градиентным затуханием и взрывом при обработке длинных последовательностей, более длительное время обучения по сравнению с другими моделями, ограниченная способность к параллельной обработке данных.

**3.3.8. Трансформерные модели**

- BERT (Bidirectional Encoder Representations from Transformers)

- GPT (Generative Pre-trained Transformer)

- RoBERTa, XLNet и другие

Преимущества: двунаправленная архитектура позволяет учитывать контекст слева и справа от слова, высокая производительность в задачах обработки естественного языка, предобучение на больших корпусах данных обеспечивает глубокое понимание языка, возможность тонкой настройки (fine-tuning) на конкретных задачах с относительно небольшими датасетами.

**3.3.9. Обоснование выбора модели BERT**

На основе проведенного анализа было принято решение использовать модель BERT для решения задачи автоматизации определения деградации в ответах HTTP-запросов. Ниже представлены основные аргументы в пользу этого выбора.

BERT специально разработан для задач обработки естественного языка и способен эффективно работать с текстовыми данными, такими как логи HTTP-запросов. Модель учитывает семантические и синтаксические особенности текста, что позволяет ей улавливать тонкие различия между нормальными и деградированными запросами.

BERT использует двунаправленный механизм внимания, который позволяет модели понимать контекст слова, учитывая как левый, так и правый контекст. Это особенно важно при анализе логов, где значение сообщения может зависеть от предыдущих и последующих слов или фраз.

Предобученная модель BERT была обучена на огромных объемах текстовых данных, что обеспечивает ей глубокое понимание языковых структур и позволяет достичь высокой точности даже при обучении на небольших специализированных датасетах.

BERT поддерживает тонкую настройку (fine-tuning), что позволяет адаптировать предобученную модель к конкретной задаче, такой как классификация логов HTTP-запросов, с минимальными изменениями в архитектуре и относительно небольшими затратами ресурсов.

Благодаря широкой поддержке сообщества и наличию мощных библиотек, таких как Hugging Face Transformers, использование BERT становится более доступным. Это облегчает процесс разработки, обучения и внедрения модели.

Многочисленные исследования и практические применения демонстрируют, что BERT достигает высоких показателей точности в различных задачах классификации текста. Это соответствует требованию высокого уровня надежности в банковской сфере.

Хотя модели на основе трансформеров требуют значительных вычислительных ресурсов, современные аппаратные решения (GPU, TPU) и оптимизированные библиотеки позволяют обеспечить приемлемую производительность. Кроме того, BERT может быть настроен для обработки данных в режиме реального времени или близкому к реальному времени.

**3.3.10. Сравнение с альтернативными моделями**

При сравнении BERT с другими моделями были выявлены следующие преимущества: превосходит традиционные методы и классические нейронные сети в способности обрабатывать сложные текстовые данные без необходимости сложной предобработки или создания признакового пространства, преодолевает ограничения рекуррентных нейронных сетей в обработке длинных последовательностей и обеспечивает более эффективную параллельную обработку данных, более эффективен в улавливании контекстных зависимостей по сравнению с моделями типа LSTM и GRU.

Выбор модели BERT для автоматизации определения деградации в ответах HTTP-запросов обоснован её соответствием ключевым требованиям задачи. Модель обеспечивает высокую точность и надежность, эффективно обрабатывает текстовые данные, учитывая контекстные зависимости, и поддерживает тонкую настройку для адаптации к специфике банковской информационной системы. Несмотря на некоторые ограничения, связанные с вычислительными ресурсами, преимущества модели BERT в значительной степени перевешивают потенциальные недостатки, делая её оптимальным выбором для решения поставленной задачи.

* 1. **Выводы о разработке и обучении модели для автоматизации определения деградаций**

В процессе разработки и обучения модели для автоматизации определения деградаций в ответах HTTP-запросов были достигнуты следующие важные результаты и сделаны соответствующие выводы. Основываясь на проведенном анализе различных методов машинного обучения и глубинных нейронных сетей, была выбрана модель BERT. Эта модель доказала свою эффективность в обработке естественного языка и выявлении сложных закономерностей в текстовых данных, что является критически важным для точного обнаружения деградаций в логах HTTP-запросов. Двунаправленная архитектура BERT обеспечивает глубокое понимание контекста, благодаря чему модель может эффективно учитывать последовательность и взаимосвязь слов в логах. Были тщательно изучены логи HTTP-запросов банковской информационной системы, и на основе этого анализа определены форматы и особенности данных. Предобработка включала очистку и нормализацию текста, токенизацию с помощью соответствующего токенизатора модели BERT, а также аннотацию логов с метками деградации для обучения и оценки модели. Такой подход к подготовке данных обеспечил высокое качество обучающего набора, что позволило повысить точность модели. Модель BERT была адаптирована под задачу бинарной классификации, что включало добавление дополнительного полносвязного слоя (классификационной "головы") на выходе модели. Были выбраны оптимальные гиперпараметры обучения, такие как скорость обучения, количество эпох, размер батча и шаг изменения скорости обучения, что позволило достичь баланса между скоростью обучения и точностью результата. Модель была обучена на подготовленном датасете с применением библиотеки transformers от Hugging Face и фреймворка PyTorch. В процессе обучения использовались функции потерь (CrossEntropyLoss) и оптимизатор (AdamW) для настройки весовых коэффициентов модели. Регулярная оценка на валидационном наборе данных позволила отслеживать метрики эффективности модели — точность (accuracy), полноту (recall), точность положительных предсказаний (precision) и F1-меру — и предотвратить переобучение за счёт ранней остановки (early stopping). По итогам обучения модель достигла высоких показателей точности и качества классификации деградированных ответов HTTP-запросов, что свидетельствует о её пригодности для реальных сценариев использования. Были выявлены наиболее значимые признаки и паттерны, которые влияют на классификацию. Это позволило лучше понять, какие аспекты логов модель использует для принятия решений. Проведенный анализ результатов обучения позволил провести тонкую настройку модели, включая корректировку гиперпараметров и стратегий обучения для дальнейшего повышения качества классификации. Итоговая модель успешно демонстрирует высокую точность определения деградаций в логах HTTP-запросов, что позволяет оперативно выявлять проблемы в работе веб-сервисов и минимизировать негативное влияние на пользователей и финансовые показатели банка. Применение модели BERT позволяет обрабатывать данные в режиме, близком к реальному времени, что является важным фактором для скорейшего реагирования на обнаруженные проблемы. Разработанная модель была успешно интегрирована в информационную систему Elastic, что позволило автоматизировать процесс мониторинга и анализа логов. Предполагается, что регулярное обновление данных для обучения модели позволит поддерживать высокую точность классификации и адаптироваться к изменяющимся условиям. В дальнейшем планируется исследовать возможности оптимизации модели, включая использование более эффективных архитектур трансформеров (например, RoBERTa, DistilBERT) или методов уменьшения размера и ускорения инференса модели.

Обобщая выводы, разработка и обучение модели для автоматизации определения деградаций в ответах HTTP-запросов продемонстрировали, что использование методов глубокого обучения на базе модели BERT обеспечивает высокую точность и надёжность классификации. Правильный выбор модели, тщательная подготовка данных, настройка гиперпараметров и анализ результатов обучения являются ключевыми факторами успеха. Полученные результаты указывают на высокую эффективность предложенного подхода для реальных условий эксплуатации в банковской информационной системе, обеспечивая тем самым своевременное обнаружение и устранение деградаций для поддержания качества обслуживания и удовлетворённости клиентов.

**4. ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ**

. . .