|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**  **«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ**  **(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»** | | | | |
| **Журнал практики** | | | | |
| Институт № 8 | «Компьютерные науки и прикладная математика» | | | |
|  |  | | | |
| Кафедра | \_\_\_806\_\_\_\_\_\_\_\_\_\_\_ | | Учебная группа | \_\_\_М8О-407Б-19\_\_\_ |
|  |  | |  |  |
| ФИО обучающегося | | \_\_Ляшун Дмитрий Сергеевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
|  | |  | | |
| Направление подготовки/  специальность | | \_\_01.03.02 Прикладная математика и информатика \_\_\_\_\_\_ | | |
|  | | *шифр, наименование направления подготовки/специальности* | | |
|  | |  | | |
| Вид практики | | \_\_\_\_\_\_преддипломная\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
|  | | *учебная, производственная, преддипломная или другой вид практики* | | |
| Оценка за практику | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Плотникова Н.О. | | |

Москва

2023

|  |  |  |
| --- | --- | --- |
| 1. **Место и сроки проведения практики:** | | |
| Наименование организации: | \_\_Кафедра 806 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |
| Сроки проведения практики |  | |
| дата начала практики: | \_\_09.02.2023\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |
| дата окончания практики: | \_\_10.05.2023\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |
| 1. **Инструктаж по технике безопасности:** | | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | / Плотникова Н.О. / | \_\_\_09 \_февраля\_\_\_ 2023г. |
| *подпись проводившего* | *расшифровка подписи* | *дата проведения* |
| 1. **Индивидуальное задание обучающегося:**   *Примечание: индивидуальное задание должно соответствовать виду (типу), целям и задачам практики (т.е. компетенциям и результатам обучения в рабочей программе практики ООП ВО)* | | |
| Разработать интеллектуальную систему для выявления аномалий сетевого трафика в локальной сети, формализовать набор параметров о сетевом трафике для выявления аномалий, проверить применимость и эффективность существующих моделей искусственного интеллекта в рамках решения данной задачи. Исходными набором данных является трёхгодовой записанный сетевой трафик компании ЦСС.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
| 1. **План выполнения индивидуального задания обучающегося:** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Место проведения** | **Тема** | **Период выполнения** |
| 1 | Кафедра 806 | Инструктаж | 09.02.23-09.02.23 |
| 2 | Кафедра 806 | Изучение и анализ существующих методов обнаружения сетевых аномалий | 10.02.23-15.02.23 |
| 3 | Кафедра 806 | Анализ сущности сетевых аномалий | 16.02.23-22.02.23 |
| 4 | Кафедра 806 | Проектирование архитектуры системы | 23.02.23-23.03.23 |
| 5 | Кафедра 806 | Программная реализация системы | 24.03.23-24.04.23 |
| 6 | Кафедра 806 | Проведение экспериментов и оценка качества работы | 25.04.23-08.05.23 |
| 7 | Кафедра 806 | Оформление журнала практики | 09.05.23-10.05.23 |

|  |  |  |
| --- | --- | --- |
| **Утверждаю** | | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | / Плотникова Н.О. / | \_\_\_09 \_февраля\_\_\_ 2023г. |
| *подпись руководителя от МАИ* | *расшифровка подписи* | *дата утверждения\** |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | / Сошников Д.В. / | \_\_\_09 \_февраля\_\_\_ 2023г. |
| *подпись руководителя от организации/предприятия* | *расшифровка подписи* | *дата утверждения\** |
| **Ознакомлен** | | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | / Ляшун Д.С. / | \_\_\_09 \_февраля\_\_\_ 2023г. |
| *подпись обучающегося* | *расшифровка подписи* | *дата ознакомления\** |

|  |  |  |
| --- | --- | --- |
| 1. **Отзыв руководителя практики от организации/предприятия:** | | |
| Студент группы М8О-407Б-19 Ляшун Дмитрий Сергеевич проходил преддипломную практику на Кафедре 806 МАИ и занимался разработкой интеллектуальной системы для выявления аномалий сетевого трафика.  В результате прохождения практики была спроектирована и программно реализована система, в основе которой были опробованы такие модели машинного обучения как случайный лес, бустинг на решающих деревьях и многослойная искусственная нейронная сеть с прямой связью. Практический эксперимент по выявлению аномалий на реальном трафике предприятия показал, что система прекрасно справляется с поставленной задачей, а алгоритмы бустинга на деревьях и многослойной искусственной нейронной сети дают правильную оценку с наибольшей точностью.  В ходе выполнения поставленной задачи Ляшун Д.С. проявил трудолюбие и старание, отличные аналитические способности и самостоятельность в принятии важных решений. Вдобавок студент показал хорошие знание различных предметных областей, в частности, машинного обучения и анализа данных, а также принципов работы компьютерных сетей.  Задачи, поставленные перед студентом за время прохождения практики, выполнены полностью и на высоком теоретическом и практическом уровне. Материалы, изложенные в отчёте обучающегося, полностью соответствуют индивидуальному заданию. | | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | / Сошников Д.В. / | \_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ 20\_\_г. |
| *подпись руководителя от организации/предприятии* | *расшифровка подписи* | *дата* |

|  |
| --- |
| 1. **Отчет обучающего по практике:** |

## 6.1 Постановка задачи

На основе данных сетевого трафика предприятия решить задачу классификации сетевых узлов. Полученная интеллектуальная система должна уметь классифицировать узлы с точностью выше 90%, в результате чего будет составлен портрет поведения устройств в сети, и при реальной работе, в случае ошибочной классификации, поведение узлов будет расцениваться как аномальное.

Итоговая система должна уметь детектировать аномалии в реальном времени и работать непрерывно.

В ходе работы необходимо составить архитектуру системы обнаружения аномалий и её программно реализовать. Также потребуется формализовать требования к набору входных данных, провести тщательный разведочный анализ имеющегося сетевого трафика, оценить наличие выбросов, сбалансированности присутствия различных сетевых узлов и т. д.

Для решения задачи предлагается использовать методы искусственного интеллекта как наиболее перспективные – необходимо провести проверку подходящих из них и выбрать наилучший.

## 6.2 Архитектура системы

Схема разработанной архитектуры интеллектуальной системы показана на рисунке 1. Рассмотрим каждый из компонентов архитектуры в отдельности.

Модуль захвата трафика – это программное обеспечение, которое используется для перехвата и записи сетевого трафика в файловом представлении. Наиболее удобным форматом представления захваченного трафика является PCAP, хранящий данные о сетевых пакетах, поскольку он имеет поддержку различных протоколов и предусматривает сжатие файлов.

Модулем преобразования трафика является набор программного обеспечения, производящего конвертацию данных о соединениях из файла PCAP в формат таблицы БД. Цель модуля преобразования трафика – облегчить анализ и мониторинг сетевого трафика для последующего модуля визуализации трафика и модуля подготовки данных. Наиболее удобным видом представления для этого является формат Netflow.

Модуль визуализации трафика позволяет пользователю увидеть на примере различных диаграмм и графиков те данные, которые в данный момент подлежат обработке в ядре системы обнаружения аномалий.

Рисунок 1 – Базовая архитектура системы обнаружения аномалий

Изображение выглядит как диаграмма, текст, План, зарисовка

Автоматически созданное описание

После заключения данных о сетевом трафике в табличную форму встаёт задача представить эту информацию в таком виде, который будет подаваться на вход моделям искусственного интеллекта – её решением занимается модуль подготовки данных.

Следующим структурным элементом выступает ядро системы обнаружения аномалий (СОА), которое состоит из модулей составления портрета и оценки. Модуль составления портрета отвечает за первоначальное обучение используемой модели искусственного интеллекта, а модуль оценки применяет итоговую обученную модель искусственного интеллекта на реальном трафике для выявления аномалий. Суть его работы заключается в попытке предсказания сетевого узла, имея только набор параметров его поведения. Если предсказание окажется неверным, то его действия расцениваются как аномальное – и система зафиксирует это событие.

Полученная итоговая архитектура интеллектуальной системы обнаружения аномалий обладает рядом преимуществ, а именно:

1. Возможность реорганизации работы каждого из модулей в отдельности, без внесения существенных правок в другие модули. Например, в ядре СОА при желании могут быть использованы любые модели искусственного интеллекта.
2. Структурные элементы системы являются исчерпывающими и покрывают любые возникающие потребности при развёртывании решения в разных условиях.
3. Система позволяет автоматизировать процесс проверки любых возникших гипотез – как качества моделей, так и значимости используемого набора параметров.

## 

## 6.3 Выбранные модели искусственного интеллекта

Для решения задачи обнаружения аномалий в сетевом трафике были выбраны следующие модели искусственного интеллекта как наиболее перспективные: случайный лес, бустинг на решающих деревьях, многослойная искусственная нейронная сеть (ИНС) с прямой связью.Схема архитектуры используемой многослойной ИНС с прямой связью показана на рисунке 2.

Рисунок 2 – Архитектура нейронной сети

Изображение выглядит как текст

Автоматически созданное описание

Слои Dense представляют структуру связи нейронов и применяют линейное преобразование к входному вектору.

Между всеми слоями Dense в нейронной сети размещаются нелинейные функции активации ReLU (rectified linear unit) для того, чтобы усложнить итоговую представляемую зависимость от входных данных. Значение ReLU вычисляется как максимум от нуля и выходного значения нейрона предыдущего слоя. Использование ReLU по сравнению с другими функциями активации позволяет быстрее и эффективнее обучать нейросеть на больших и сложных наборах данных.

Поскольку данные на вход нейронной сети подаются группами, называемыми батчами, слои Batch Normalization отвечают за их нормализацию – центрирование и повторное масштабирование. Это делается с целью более быстрого и стабильного обучения сети.

В предложенной архитектуре многослойной ИНС с прямой связью также задействуются слои Dropout. Они используют подход регуляризации для уменьшения избыточного переобучения путём случайных занулений выходов нейронов в ходе построение модели.

Выход последнего полно связного слоя является вектором с размерностью равной общему числу классов, который далее преобразуются функцией активации Softmax для нормализации полученных значений к вероятностям принадлежности к каждому из классов сетевых узлов.

Полученный вектор с вероятностями принадлежности каждому классу является итоговым выходом модели. Индекс максимального значения в нём можно расценивать как метку того класса, которому соответствуют поступившие входные данные о сетевом трафике.

## 6.4 Стек используемых технологий

Программирование системы и реализация моделей искусственного интеллекта осуществлялось на языке Python3 c использованием таких библиотек как:

* Pandas – для обработки и анализа структурированных данных;
* Keras, Tensorflow – для реализации моделей глубокого обучения;
* Matplotlib, Plotly – для визуализации данных двумерной и трёхмерной графикой;
* Numpy – для линейной алгебры и математических вычисления над матрицами;
* Imblearn – для проведения балансировки исходного набора данных;
* XGBoost, CatBoost – для использования моделей градиентного бустинга.

Также в ходе разработки интеллектуальной системы были задействованы следующие программы и утилиты:

* Clickhouse – для представления данных о сетевом трафике в виде БД;
* Nfdump – для обработки сетевого трафика в формате Netflow;
* Tcpdump & libpcap – для захвата сетевого трафика и последующего хранения в виде файлов формата PCAP.

## 6.5 Описание программной разработки

Решение программировалось в среде Jupyter Notebook – каждый из блоков кода в ноутбуке отвечает за свою задачу в рамках описанной ранее модульной структуры, и вывод результата работы любого блока представлялся непосредственно в ноутбуке в виде графического (гистограммы распределения узлов, важности признаков и т. д.) или же текстового представления (предсказания модели, оценки качества).

Первоначальный захват трафика осуществляется с помощью утилиты Tcpdump. Используя встроенную библиотеку Libpcap, она конвертирует проходящий через неё трафик в файловый формат PCAP, который затем хранится на рабочем сервере.

На следующем шаге работы системы модуль подготовки данных производит преобразование имеющихся файлов PCAP в представление Netflow. Для этого вызывается специальная подпрограмма, в основе которой лежит утилита Nfpcapd, являющаяся частью более общего решения – Nfdump.

В ходе её работы производится отбор нужных для преобразования файлов PCAP путём сравнения их временных меток модификации, после чего выполняется преобразование в формат Netflow методом Nfpcapd – c уточнённым набором параметров запуска, в результате чего сетевой трафик нарезается на минутные интервалы и раскидывается по соответствующим файлам. Далее вызывается утилита Nfdump, которая представляет данные файлов Netflow в удобном для дальнейшего чтения виде – формате CSV. Итоговые файлы CSV используются для формирования результирующей таблицы БД Clickhouse.

Описание набора параметров сетевых потоков Netflow в представлении таблицы БД Clickhouse показано в таблице 1.

Таблица 1 – Состав таблицы Clickhouse с характеристиками сетевого трафика

| Название параметра | Тип данных | Описание |
| --- | --- | --- |
| id | UUID | Идентификатор записи |
| tsstart | UInt32 | Время регистрации первого пакета потока в секундах (Unix timestamp) |
| tsend | UInt32 | Время регистрации последнего пакета потока в секундах (Unix timestamp) |
| duration | Float64 | Длительность потока в секундах и миллисекундах |
| prot | UInt8 | Протокол (TCP, UDP, ...) |
| sip | String | IP-адрес источника |
| sport | UInt32 | Порт источника |
| dip | String | IP-адрес приёмника |
| dport | UInt32 | Порт приёмника |
| flags | String | TCP-флаги |
| pkts | UInt64 | Количество пакетов в потоке |
| bytes | UInt64 | Количество байт в потоке |
| bps | UInt64 | Среднее количество байт/сек в потоке |
| pps | UInt64 | Среднее количество пакетов/сек в потоке |
| bpp | UInt64 | Среднее количество байт/пакет в потоке |

После заключения данных о сетевом трафике в табличном представлении его можно визуализировать посредством методов модуля визуализации трафика. Для построения графиков с числом записей и встречаемости каждого узла в сети достаточно сделать запросы в таблицу Clickhouse на выдачу количества записей по каждому IP и полученную информацию использовать для отрисовки соответствующих гистограмм методами библиотеки Matplotlib языка Python. Визуализация поведения сетевых узлов в виде графика параллельных координат выполняется с использованием библиотеки Plotly. При построении графиков из таблицы Clickhouse извлекаются данные о сетевых соединениях конкретного узла в рамках одного дня, которые затем фильтруются по протоколу сетевого уровня и агрегируются в пятиминутные промежутки – с целью уменьшения расходов на вычислительные ресурсы.

На следующем этапе работы системы модуль подготовки данных извлекает параметры сетевого трафика из таблицы БД Clickhouse путём запросов на выборку по временному промежутку и требуемому IP. При получении очередной порции данных проверяется отсутствие дубликатов, пропусков, а также производится бинаризация флагов TCP из строкового представления. Финальным шагом является агрегация по установленному временному окну.

Алгоритм агрегации для формирования записи устроен следующим образом:

1. Задаётся частота запуска расчёта – например, 1 минута.
2. Задаётся исторический интервал, в рамках которого формируется итоговая запись – например, 10 минут.
3. Задаётся количество промежутков исторического интервала.
4. Исторический интервал делится на промежутки по установленному количеству. Например, в случае настройки поделить 10 минутный интервал на 5 частей будут получены промежутки c минутами 1-2, 3-4, 5-6, 7-8, 9-10 соответственно.
5. Записи о соединениях размещаются по соответствующим временным промежуткам исторического интервала.
6. В рамках каждого промежутка происходит агрегация параметров – вычисление средних величин для скорости передачи пакетов и байт; cуммы для числа переданных или полученных байт и пакетов; числа уникальных портов, протоколов, или удалённых хостов.
7. Итоговый набор просчитанных параметров является единой записью для последующей обработки моделями.

Сформированная по представленному алгоритму запись охватывает больший временной промежуток, чем обычное соединение, тем самым заметно обобщая поведение сетевого узла.

Описанная ранее запись с историческим окном имеет время конца, которое представляется в другом, отличном от Unix timestamp формате – циферблатном времени, вычисляемом по формуле (1):

где – число часов от начала текущего дня;

– число минут от начала текущего часа.

Циферблатное время указывает на косинус и синус углов часовой и минутной стрелок. Использование обоих тригонометрических функций необходимо с целью однозначного определения времени.Представление в циферблатном формате призвано сделать временные значения более правильными – периодическими, чтобы модели могли лучше запоминать и распознавать повторяющиеся с течением времени события.

Также к циферблатному времени добавляется другие временные параметры – день недели и время суток, которое может принимать следующие значения:

1. утро – от 6 до 12 часов;
2. день – от 12 до 16 часов;
3. вечер – от 16 до 24 часов;
4. ночь – от 24 до 6 часов.

На основе параметров Netflow происходит создание более сложных по смыслу агрегационных параметров – отбираются наиболее активные сервисы и все протоколы сетевого уровня, которые встречались, после чего по каждому из них может вычисляться:

1. среднее по продолжительности передачи данных, числу байтов в секунду, пакетов в секунду, байтов в пакете;
2. сумма по числу соединений, переданных и полученных байтов, пакетов;
3. количество различных портов, узлов приёма или получения данных.

Выбор этих параметров варьируется от результатов отбора по значимости, в случае же первоначального развёртывания системы потребуется просчёт всех из них. Также стоит отметить, что агрегационные параметры определяются в рамках своего промежутка исторического окна, а также с учётом входящих и выходящих соединений.

При создании обучающего набора данных сперва задаётся ограничение на максимально возможное число записей по каждому сетевому узлу в выборке – это первый шаг корректировки несбалансированности. На втором шаге осуществляется целевая выборка тех классов-IP, которые существенно преобладают над остальными. Происходит это по принципу квантиля – задаётся процентный порог от всего объёма данных, после чего выполняется последовательный отбор классов-IP со всеми соответствующими им записями, начиная с наиболее встречаемых классов, до того момента, когда доля набранных записей от общего объёма превосходит процентный порог. В рамках решения задачи значение квантиля составляло 95%. Оставшиеся классы-IP, которые не вошли в целевую выборку, приравниваются к классу неизвестный сетевой узел.

Последним шагом корректировки несбалансированности является использование технологии SMOTE. Synthetic Minority Over-sampling Technique основана на создании синтетических примеров для класса, который имеет меньшее количество образцов, чтобы тем самым уравновесить количество образцов в каждом классе.

Как было отмечено ранее, ядро СОА состоит из модуля составления портрета и модуля оценки. Перед началом работы модуля оценки модуль составления портрета должен оценить сформированный набор параметров сетевого трафика на значимость. Для этого берётся легковесная модель случайного леса, которая подвергается многократному обучению на разных подмножествах признаков с задействованием технологии оптимизации RFECV. Recursive Feature Elimination with Cross-Validation – это метод автоматического отбора признаков, который позволяет определить наиболее значимые признаки для построения модели.

После получения набора наиболее значимых признаков производится обучение итоговых моделей искусственного интеллекта. Реализация модели случайного леса взята из библиотеки Scikit-learn языка Python, модели градиентного бустинга – из библиотек XGBoost и CatBoost Программная реализация архитектуры нейронной сети с использованием методов Tensorflow представлена в репозитории проекта на Github.

При необходимости в ходе обучения моделей могут быть задействованы различные методы оптимизации по гиперпараметрам, одним из таких методов является RandomizedSearchCV.

## 6.6 Обучение моделей

Для обучения моделей искусственного интеллекта была сформирована выборка из реального сетевого трафика предприятия, которая имела верхнее ограничение на число записей по каждому сетевому узлу равное ста тысячам за промежуток 23.06.2023 – 23.09.2023. Всего имеется 135 классов наиболее активных устройств, для которых был проведён отбор значимых признаков и последующая балансировка методом SMOTE с дальнейшим обучением моделей искусственного интеллекта. Полученные результаты точности классификации сетевых узлов на тестовой выборке представлены в таблице 2.

Таблица 2 – Точности распознавания поведения сетевых узлов на тестовой выборке

| Название модели | Первоначальная точность | Точность после оптимизации по гиперпараметрам |
| --- | --- | --- |
| Случайный лес | 45% | 57% |
| CatBoost | 68% | 84% |
| XGBoost | 77% | 90% |
| Многослойная ИНС с прямой связью | 91% | 91% |

В качестве метрики оценки качества здесь используется сбалансированная точность, которая вычисляется как среднее арифметическое между точностью и полнотой для каждого класса.

Для оценки качества обнаружения аномалий в имеющийся трафик были намеренно добавлены различные сетевые угрозы: DoS-, DDoS-атаки, деятельность червей, отключения и другие. Результаты классификации полученного трафика на аномальный и неаномальный представлены в таблице 3.

Таблица 3 – Точности детектирования аномалий при различных сетевых атаках

| Название модели | Случайный лес | CatBoost | XGBoost | Многослойная ИНС с прямой связью |
| --- | --- | --- | --- | --- |
| Тестовая выборка | 21816 | | | |
| Число обнаруженных атак (всего 14527) | 7389 | 10754 | 13684 | 13932 |
| Количество записей, оценённых как неаномальный (всего 7289) | 4325 | 5252 | 6438 | 6987 |

## 6.7 Заключение

Можно сделать вывод, что модель градиентного бустинга XGBoost и многослойной ИНС с прямой связью являются наиболее эффективными для решения задачи распознавания сетевых узлов, в то время как модель случайного леса и градиентного бустинга CatBoost показали себя далеко не лучшим образом, и их точность распознавания ниже удовлетворимого уровня. Также стоит добавить, что модели градиентного бустинга XGBoost и многослойной ИНС показывают хороший результат в разграничении аномального и неаномального сетевого трафика.

|  |  |  |
| --- | --- | --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | / Ляшун Д.С. / | 10 мая 2023 г. |
| *подпись обучающегося* | *расшифровка подписи* | *дата* |