**СЛАЙД 1**

Добрый день

Меня зовут Разумов Тимофей

Моя работа посвящена применению методов машинного обучения для решения задачи фильтрации нежелательных данных

В данной работе рассмотрены различные методы предобработки, векторизации и классификации текста. А также разработано высоконагруженное приложение имеющее микро-сервисную архитектуру

**СЛАЙД 2**

в настоящее время, объемы производимой человечеством информации

увеличиваются в геометрической прогрессии.

многие мошенники активно пользуются этим, например предпринимают попытки украсть пользовательские данные или нелегально распространить рекламу используя массовые рассылки в электронной почте

**СЛАЙД 3**

постановка

**СЛАЙД 4,5,6**

первым этапом нужно осуществить предобработку данных. мы имеем дело с электронными письмами. то, что видит пользователь в web интерфейсе существенно отличается от исходного формата письма RFC.

как видно, сырой формат имеет очень много служебных заголовков, которые не несут никакой смысловой нагрузки для анализа данных.

нас интересуют только заголовки subject (тема письма) и body (содержимое).

**СЛАЙД 7**

в самом письме может быть как текстовая информация: html, calendar, текст, так и графическая: pdf, изображения.

для каждого из текстовых форматов реализованные специализированные парсеры на c++ и golang.

для извлечения текста из изображений и pdf разработаны две нейронные сети на языке python. первая - зондирования и сегментации, вторая - модель извлечения текста

**СЛАЙД 8**

после извлечения текста из всех вложений письма происходит его конкатенация с разделяющими символами, затем декодирование в общую кодировку, чаще всего это cp1251 или utf8.

затем производится удаление лишних пробелов, отступов и стоп-слов, которые спамеры часто используют для зашумления.

далее производится нормализация слов в тексте

стемминг - грубый процесс отбрасывающий окончания и лемматизация - более тонкий процесс, использующий морфологический анализ для приведения слов к нормальной форме, но занимающий больше вычислительных ресурсов. Поскольку лемматизация дает наиболее точные результаты, то именно этот подход используется для дальнейших расчетов

**СЛАЙД 9,10**

после полного извлечения текста из письма требуется построить отображения текста в векторное пространство.

мчитаем, что ….

**СЛАЙД 11**

простейший метод векторизации текста предложенный в начале 2000 это мешок слов.

….

**СЛАЙД 12**

в 2016 г. эта проблема была решена Фейсбуком при помощи word2vec модели. В ней ….

**СЛАЙД 13**

существует оптимизация этого алгоритма, которая заключается в том, что …

где в каждом листе дерева закодировано слово

с помощью такой оптимизации удается снизить сложность до log2 N, так как дерево Хаффмана присваивает короткие коды частым словам.

**СЛАЙД 14**

мы построили отображение из текстового пространства в векторное. теперь нужно по векторизованным текстам научиться относить письмо к спаму или нет

**СЛАЙД 15**

формально …

**СЛАЙД 16**

эта задача эквивалентна …

**СЛАЙД 6**

Поскольку современные алгоритмы машинного обучения не могут напрямую работать с сырым текстом, необходимо построить отображение текста в векторное пространство. Рассмотрим несколько подходов для построения данного отображения

**СЛАЙД 7**

Модель "мешок слов" - это упрощенное представления текстовой информации, используемое в задачах обработки естественных языков. В этой модели текст представляется в виде мультимножества его слов

**СЛАЙД 8**

Основной проблемой мешка слов является потеря контекста между словами. Поскольку в естественном языке перестановка даже двух слов предложения может полностью изменить его смысл, то данный подход к классификации может иметь низкую точность.

Word2Vec инструмент векторизации текста с учетом контекстной связи между словами.

Алгоритм построен из следующих шагов. Сначала

1. Чтение текста, и
2. …..

FastText - это расширение Word2Vec, предложенное Facebook в 2016 году. Вместо ввода отдельных слов в нейронную сеть, FastText разбивает слова на несколько n-грамм (подслов). Вектор для слова "apple" образуется при помощи суммы всех n-грамм. После обучения нейронной сети у нас будут эмбеддинги слов для всех n-грамм из обучающего набора данных.

С помощью данного подхода алгоритм становится более чувствительным к редким словам так как весьма вероятно, что некоторые из их n-грамм также присутствуют в других словах.

**СЛАЙД 9**

После векторизации текста мы должны построить преобразование из множества признаков во множество классифицируемых объектов

**СЛАЙД 10**

Наиболее частым методом используемым в задаче классификации используется метод логистической регрессии:

В таблице приведены результаты расчетов для разных способов векторизации текста, из которых видно, что FastText модель лучше справляется со своей задачей

**СЛАЙД 11**

Также в работе была рассмотрена более глубокая нейронная сеть, имеющая три слоя. Где на первом и втором слое используется функция активации ReLu, а на последнем слое сигмоида. При обучении для поиска весов используется метод Адама с регуляризацией Тихонова

В таблице также приведены результаты расчетов для разных способов векторизации текста, из которых видно, что на полносвязной сети получилось достичь более высокую точностью

**СЛАЙД 12**

Поскольку сервис должен работать под высокими нагрузками важно обеспечить архитектуру, при которой выход из строя одной компоненты не приведет к деградации всей системы

**СЛАЙД 13**

Реализована следующая клиент-серверная архитектура (рис. 1). Antispam daemon (mrasd) парсит входящие сообщение и извлекает оттуда текст, изображения, файлы. Поскольку нежелательные данные часто содержатся внутри вложенных изображений и документов, то из них также необходимо извлечь текст. Для этого через отложенную redis очередь документы оправляются в OCR сервис, который извлекает текст и сохраняет результат в redis cache.

Так как, с большой вероятностью письмо может быть дубликатом (например в случае ddos атаки или оффлайн перепроверки), то чтобы не нагрузать лишний раз OCR сервис, производится первичная проверка redis cache на наличие уже обработанных данных по заданному хэшу документа.

После полного извлечения текста сервис mrasd производит все этапы предобработки текста (приведение к одному регистру, удаление стоп слов, нормализация), а затем отправляет текст в сервис mlapi по протоколу grpc для векторизации текста при помощи fast text и дальнейшего получения предсказания fcnn модели по котому принимается решение о "нежелательности" входящего сообщения.

Предложенная архитектура хороша тем, что при выходе из строя OCR сервиса, одного (или нескольких) инстансов redis сluster, не деградирует вся система в целом.

**СЛАЙД 14**

на этом слайде отображено среднее количество запросов в минуту и среднее время обработки запроса. если их проанализировать будет видно, что даже в пики при 10-ти часовой рассылке система продолжает хорошо себя чувствовать и работать с прежней эффективностью