Автономная некоммерческая общеобразовательная организация "Физтех-лицей"
(АНОО «Физтех-лицей» им. П.Л. Капицы)

XX научно-практическая конференция

«Старт в инновации»

Clever Rating

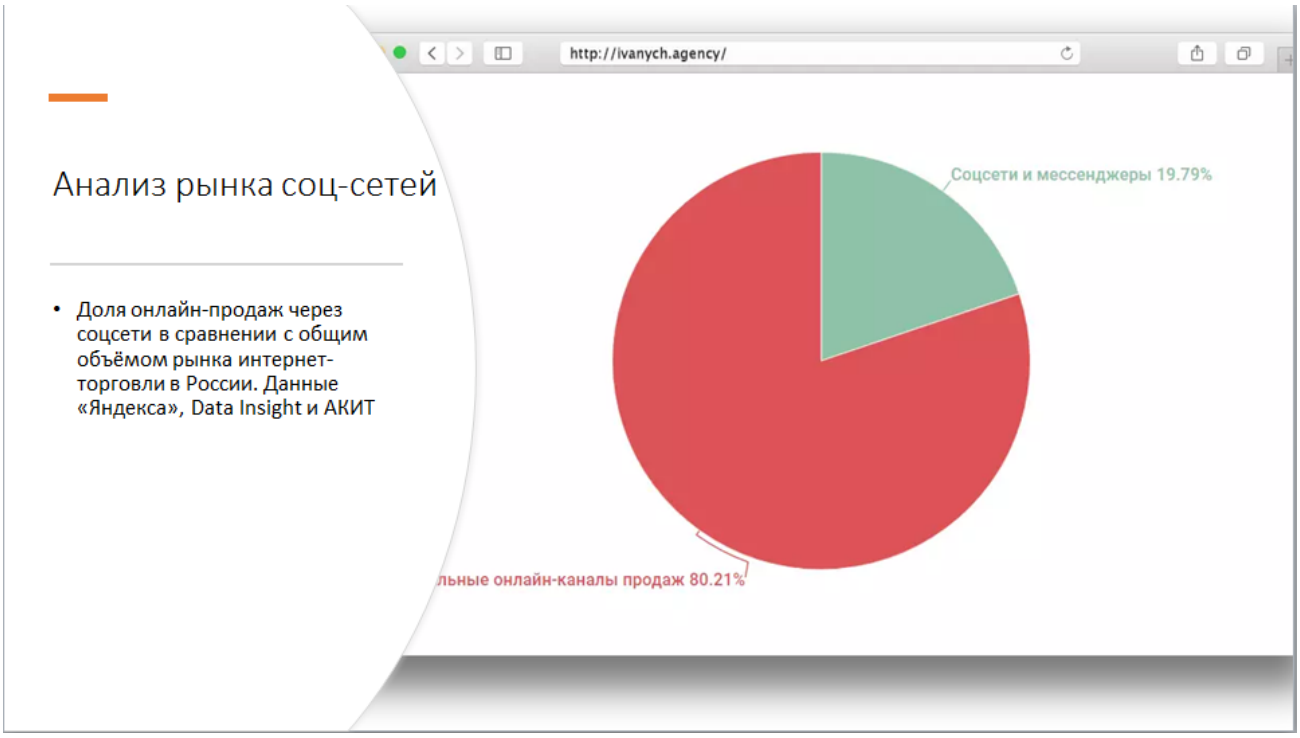
Выполнили:
Кочкарев Иван
Суслов Матвей
Вольнов Сергей
Руководитель:
Аминов Тимур

Московская область, г. Долгопрудный

2021 г.

Введение:

В последнее время социальные сети занимают все большую часть нашей жизни. Мы стали настолько неразлучны со своей техникой, что статус «всегда онлайн» уже никого не удивляет. И, конечно же, это повлияло и на экономику. Согласно проведенному в 2018 году исследованию Яндекса совместно с Data Insight и АКИТ, уже тогда доля рынка, приходящаяся на социальные сети достигала 20% от всех интернет продаж. И этот рынок не перестает развиваться. Поэтому мы с командой решили поближе его рассмотреть.



При более подробном изучение, мы поняли, что этот сектор рынка имеет свои слабые стороны, которые можно улучшить. А именно социальные сети плохо приспособлены для осуществления продаж, не смотря на то, что многие группы этим занимаются. В отличие от Яндекс Маркета и других полноценных интернет магазинов, у мессенджеров не развита система отзывов и обратной связи. При осуществлении покупки единственное место, откуда ты можешь получить информацию о товаре от людей, которые уже приобрели его, это читать огромное количество отзывов, которых под постом может быть до нескольких тысяч.

В связи с этим мы решили разработать сервис, способный анализировать комментарии под постом, предсказывать рейтинг, который бы поставил покупатель тому или иному продукту, если бы у него была такая возможность (как, например, ставить звездочки в Яндекс Маркете). И после этого предсказывать общий рейтинг товара.

Другими словами, мы будем проставлять те самый звезды рейтинга для товаров в социальных сетях.

Цель: разработать нейросетевой алгоритм, для оценки товара по текстам отзывов, а также создать сайт, для возможности использования алгоритма обычными пользователями.

Задачи :

1) Собрать базу данных для обучения нейросети
2) Написать нейросеть и обучить алгоритм
3) Написать парсер комментариев для ВК
4) Разработать сайт
5) Провести тестирования нашего сервиса

Целевая аудитория :

1) Пользователи соц сетей, осуществляющие покупки на данной платформе
2) Владельцы групп, которые будут использовать сервис для расширения функционала своих групп
3) Маркетологи, которым необходимо получать feedback по своим товарам, даже если те продаются не в интернет-магазинах

Ход работы

Сбор данных

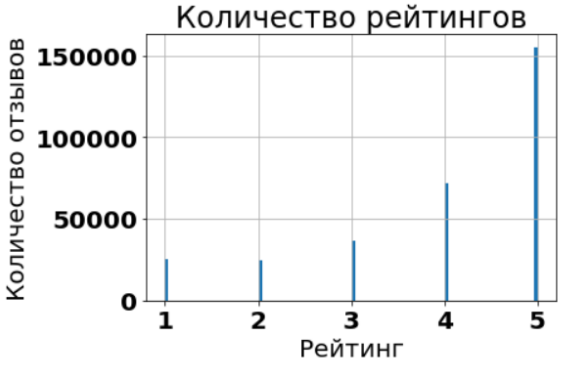
В качестве данных для обучения модели мы использовали данные с Яндекс Маркета. Так как данных было очень много мы решили предсказывать оценку товара только для телефонов и планшетов.

	prod_id	price	brand	id	author	description	rate	date	mean_price
0	10692976.0	3085.57234	23.0	280	Konstantin Zeleznov	достоинств больш экра карт min быстродейств фу...	4	2014-03-04	3085.57234
1	10692976.0	3085.57234	23.0	281	Юлия Л.	достоинств очен плох угл обзор записыва плюс с...	5	2014-03-22	3085.57234
2	10692976.0	3085.57234	23.0	282	Имя скрыто	достоинств 5 дюймов экра android вход 2 отличн...	5	2014-05-18	3085.57234
3	10692976.0	3085.57234	23.0	283	роман п.	достоинств давольн неприхотлив модел удобн про...	5	2014-03-28	3085.57234
4	10692976.0	3085.57234	23.0	284	Александр Н.	достоинств экра 5 процессор справля ур флэш 32...	4	2014-02-24	3085.57234

Тут мы столкнулись с несколькими проблемами :

1) Несмотря на то, что мы и так работали только с одной категорией товаров, у нас было порядка 300к отзывов + они были достаточно объемные. Так как мы были ограничены мощностями google colab (сервис от гугла, на котором можно обучать нейронки, так как они предоставляют свои GPU), то не могли обработать столько информации.

2) После более подробного рассмотрения наших данных мы обнаружили, что она не сбалансирована (примеры по категориям распределены не равномерно)



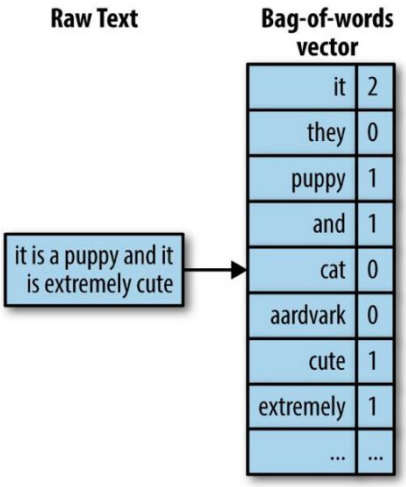
Решили мы их следующим образом : взяли по 4000 отзывов для каждого рейтинга, таким образом мы сбалансировали данные и сделали их адекватное кол-во, чтобы можно было обучать модели.

Прежде чем мы рассмотрим алгоритмы, применяемые нами для решения этой задачи, хочу ознакомить вас с несколькими стандартными методами работы с текстами.

Методы работы с текстами

1) Bag of words (BOW)

   Суть этого подхода заключается в том, чтобы сопоставить каждому тексту массив, где на arr[i] означает сколько раз слово под номером i встретилось в нашем тексте. Обычно поступают следующим образом : составляют словарь всех слов, которые использовались хотя бы в одном тексте, дальше сортируют все слова по частоте встречаемости и дальше для каждого отдельного текста (в нашем случае отзыва) считают сколько раз оно встретилось.



2) TF-ID

   Это усовершенствованный BOW. Здесь TF и IND – это показатели слова (TF — term frequency, IDF — inverse document frequency). Они несут в себе больше информации и следовательно показывают лучшие результаты при предсказаниях.

$$TF(\boldsymbol{term}) = \frac{Number\ of\ times\ \boldsymbol{term}\ appears\ in\ a\ document}{Total\ number\ of\ items\ in\ the\ document}$$

$$IDF(\boldsymbol{term}) = \log\left(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ \boldsymbol{term}\ in\ it}\right)$$

3) Word2vec

Здесь люди зашли еще дальше. Основная идея этого метода заключается в том, чтобы представить слову как точку в n-мерном пространстве. То есть мы каждому слову сопоставляем вестор длины n. Сами значения вектора могут менятся в процессе обучения, что и делает этот метод настолько универсальном. К примеру, word2vec обученный на Википедии настолько хорошо отражает грамматические и семантические связи языка, что queen – king = woman – man

Написание нейросети

Прежде всего нам необходимо привести в порядок данные, для этого мы использовали модуль nltk и библиотеку pymorphy. С помощью них мы удалили различные знаки препинания, привели все к нижнему регистру, а так же каждое слово заменили его стандартной формой (думаешь -> думать)

Для решения данной задачи мы использовали различные подходы, давайте рассмотрим их подробней

1) Методы машинного обучения

Для начала мы решили попробовать что-нибудь простое, чтобы было от чего отталкиваться. Мы обучили модель линейной регрессии на TF-IDF наших отзывов. Это дало примерно MAE (Mean absolute error) = 1.2

То есть другими словами в среднем наши предсказания отличались от правильного на 1.2 пункта. Так как всего различных вариантов рейтинга было 5, то мы решили попробовать другие алгоритмы.

2) Full-connection

Здесь мы использовали несколько полносвязных слоев в связке с TF-IDF, но это не сильно помогло MAE на валидационной выборке все равно остался около 1.2

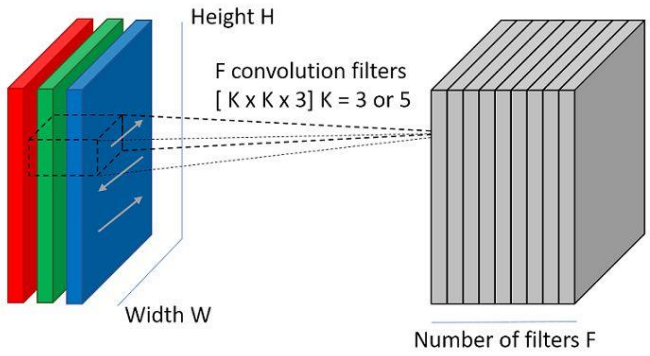
3) Convolution + embedding

В этот раз мы уже начали использовать технологию word2vec (=embedding). Так как всю нашу последовательность слов можно представить в виде двумерного массива, где каждый столбец – это его представление в n-мерном пространстве, то нам пришла идея использовать свертки для предсказания.

Принцип работы сверточных нейросетей

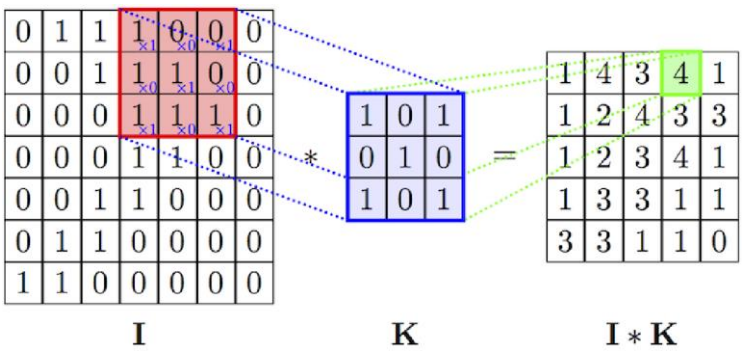
Если рассматривать их применения для изображений, то архитектура нашей сети будет выглядеть как ансамбль слоев светки (для выделения признаков) и слоев пулинга (для уменьшения размерности изображения)

Сверточный слой



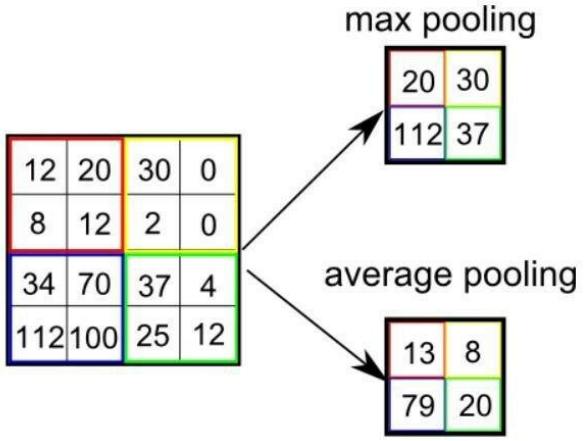
Данный слой основана на понятие свертки. Давайте рассмотрим это по подробней. Сама происходит следующим образом :

1) Фиксируем ядро свертки
2) Проходимся нашим окошком по всем каналам предыдущего слоя (в случае, когда предыдущий слой – это входное изображение, каналами являются RGB, а сами значения – это матрица, где в каждой ячейки лежит интенсивность пикселя по данному цвету)
3) Делаем поэлементное умножения свертки на рассматриваемый участок изображения
4) Суммируем ячейки, полученные после умножения
5) Записываем эту сумму в новую карту признаков



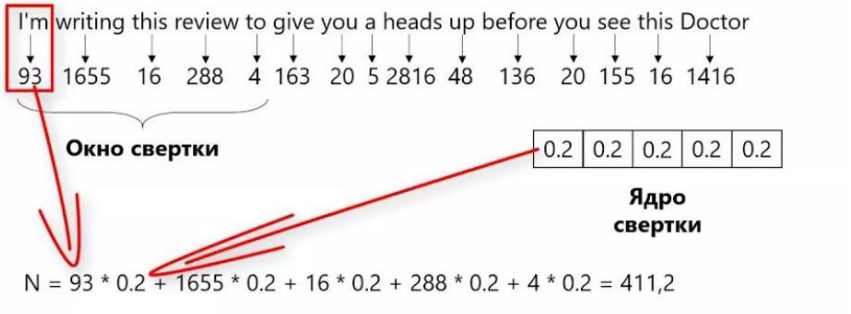
Слой подвыборки (пуллинг слой)

Здесь все еще проще. Задачей этого слоя является уменьшение размеров карт признаков, чтобы оставить только важную информацию и была возможность рассматривать изображение на макро уровне. Всего есть два основных типа этого слоя max pooling – выбираем максимальный элемент из рассматриваемого окна и average pooling – берем среднее.



Свертки в текстах

Схожий подход мы применили и в нашей задачи, только теперь свертка была одномерное (а не двумерной, как в случае с изображениями) : у слова есть n параметров (вспомним word2vec) и для каждого из них (параметров) сделаем следующую операцию :

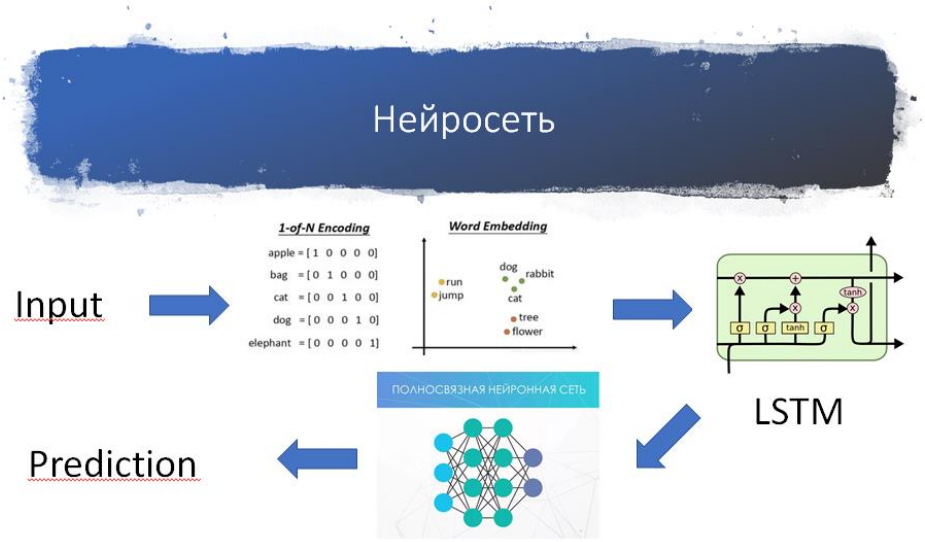


Таким образом у нас будет n каналов и для каждого из них мы будем делать свертки.

В нашем случае мы решили немного усовершенствовать этот подход и мы применяли свертки нескольких различных размеров, а потом склеивали результаты их работ. Этот метод дал результаты уже лучше, MAE = 0.9. Но мы не остановились и продолжили пробовать различные архитектуры.

4) LSTM + embedding

В данном подходе для распознавания рейтинга использовали рекуррентные нейронные сети, а именно LSTM. В начале наши данные преобразуются в embedding, затем обрабатываются с помощью рекуррентной ячейки LSTM, а после этого подаются на полносвязный слой, где и делается окончательный prediction. Такой метод выдал точность MAE = 0.8 (УРАААА)

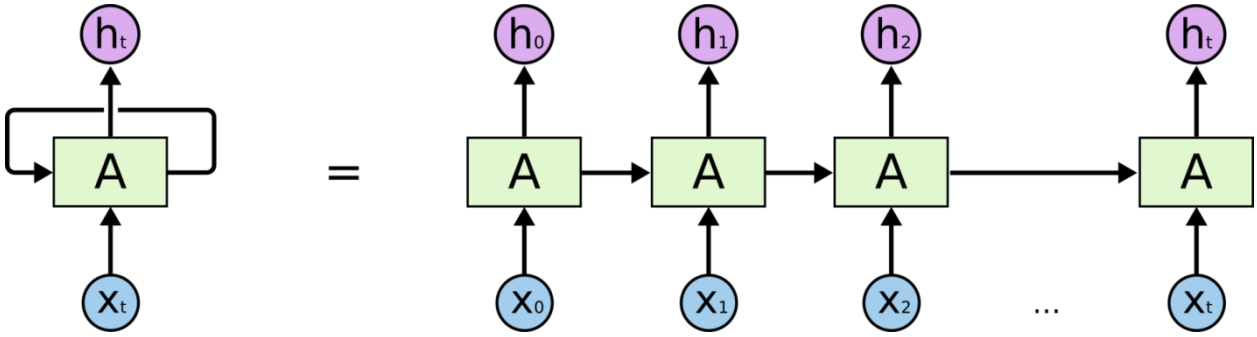


Рекуррентные сети (RNN) и LSTM

Основное преимущество таких сетей является их способность запоминать информацию, в отличие от классических НС. Благодаря этому данный тип отлично подходит для анализа различных последовательностей : от видеоряда, до предсказания курса акций по истории его курса.

Рекуррентные сети содержат обратные связи, за счет которых и создается их «память».

Рекуррентную сеть можно рассматривать, как несколько копий одной и той же сети, каждая из которых передает информацию последующей копии. Вот, что произойдет, если мы развернем обратную связь:



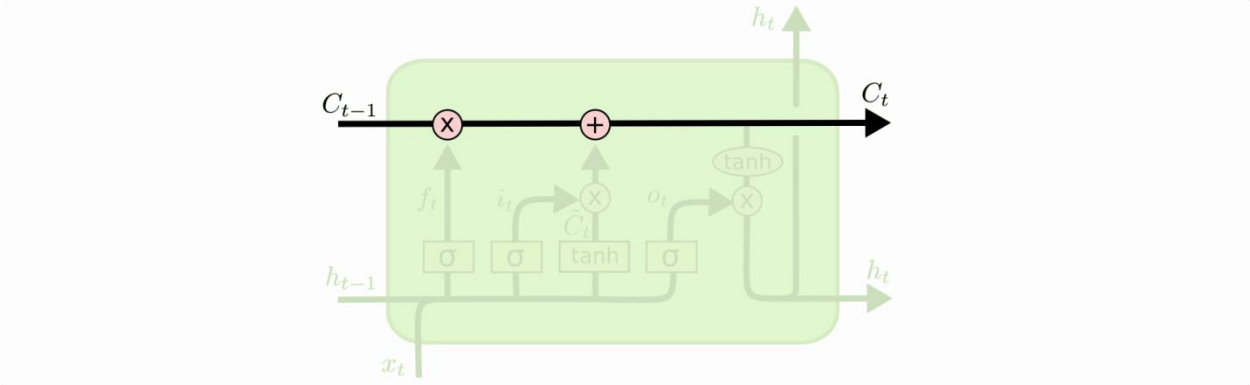
Проблема долговременных зависимостей

Одна из привлекательных возможностей RNN состоит в возможности запоминать информацию. Но, к сожалению, тут не все так гладко. Обычные рекуррентные сети могут запоминать информацию только с нескольких последних состояний, то есть они обладают кратковременной памятью, а долговременная у них отсутствует. Рассмотрим для примера 2 предложения, каждое из которых нужно закончить каким-то словом. Первое : «облака летели по ___», в данном случае слово «облака» находится недалеко от слово, которое необходимо предсказать и здесь наша сеть справится легко. Однако, если предложение будет что-то вроде «Он родился во Франции… Мальчик бегло говорит по ___». На этот раз необходимая для предсказания информация находится далеко от слова, поэтому к тому моменту, как нам придется предсказывать мы уже забудем, что там было в начале, так как к сожалению, по мере роста этого расстояния, RNN теряют способность связывать информацию.

Сети LSTM

Долгая краткосрочная память (Long short-term memory; LSTM) – особая разновидность архитектуры рекуррентных нейронных сетей, способная к обучению долговременным зависимостям. Сама сеть имеет форму цепочки из повторяющихся модулей. Но здесь, в отличие от обычной RNN, устройство модуля намного сложнее.

Ключевой компонент LSTM – это состояние ячейки (cell state) – горизонтальная линия, проходящая по верхней части схемы.



Состояние ячейки напоминает конвейерную ленту. Она проходит напрямую через всю цепочку, участвуя лишь в нескольких линейных преобразованиях. Информация может легко течь по ней, не подвергаясь изменениям.

Для более подробного изучения LSTM можете обратиться к списку литературы.

Написание парсера

Комментарии мы брали из сообщества Вконтакте. Для этого использовали библиотеку vk api

VK_api

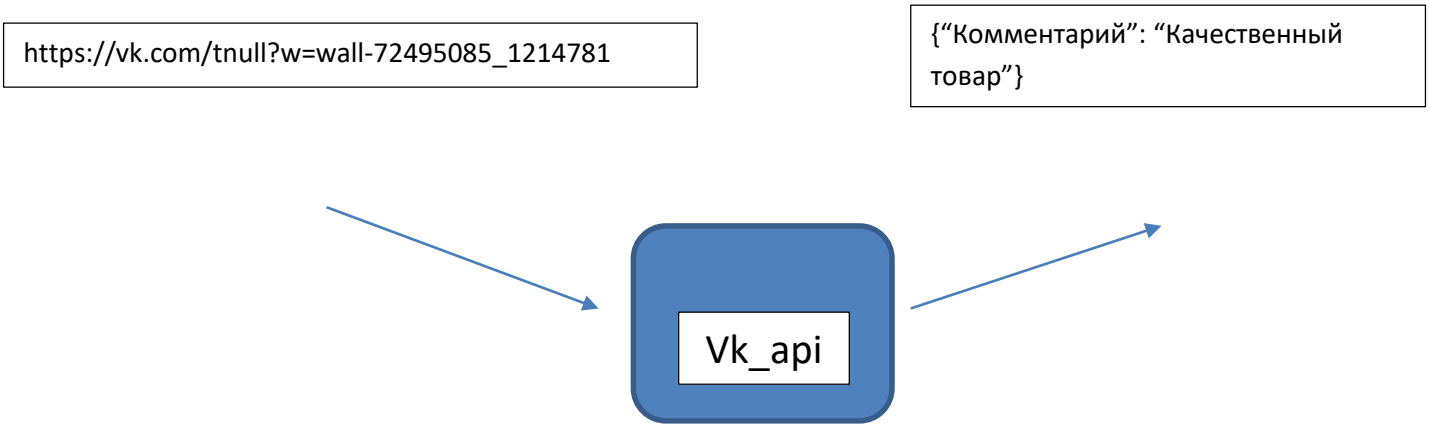
Эта библиотека позволяет получить доступ к функционалу ВКонтакте. С помощью неё мы получаем комментарии под постом и после их обработки отправляли на вход нейросети. Для работы с библиотекой vk_api мы использовали python.

Токены и авторизация

Vk_api использует для авторизации пользователей систему токенов. Каждый токен обладает своим уровнем доступа. Если токен предоставляет полные права доступа, то через vk_api можно полноценно управлять страницей.

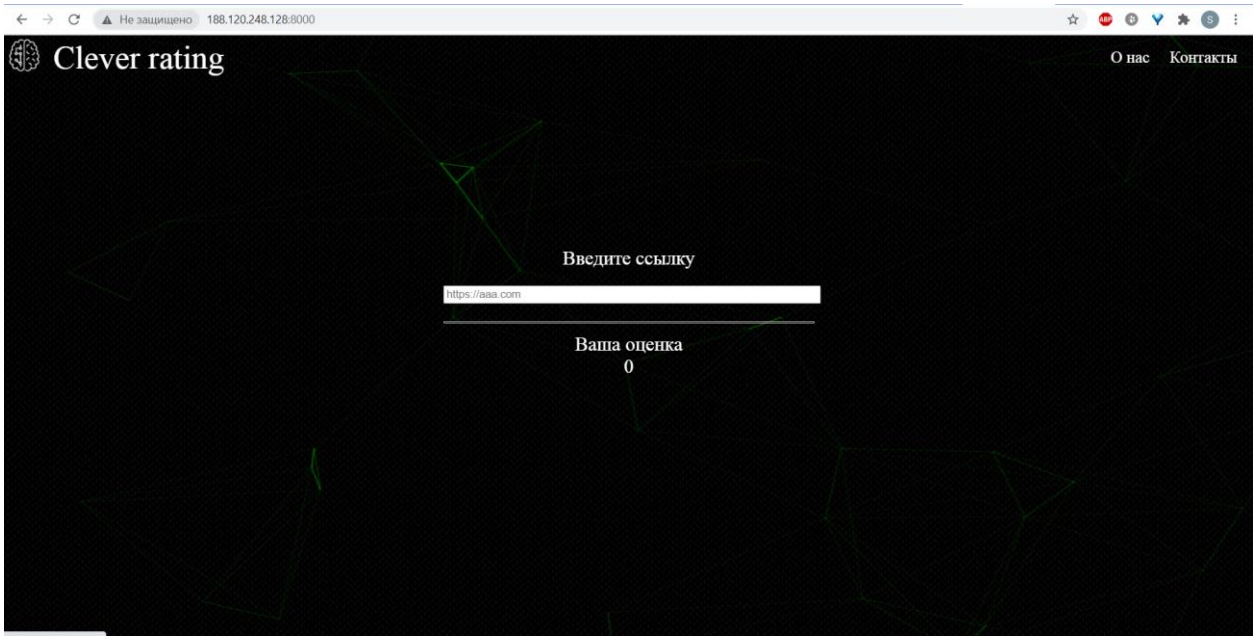
Работа с библиотекой

1. На вход программе поступает ссылка
2. С помощью регулярного выражения мы проверяем ссылку на корректность и вытаскиваем идентификатор страницы и поста на ней
3. Методом wall.getComments мы получаем комментарии к посту в формате JSON



Разработка сайта

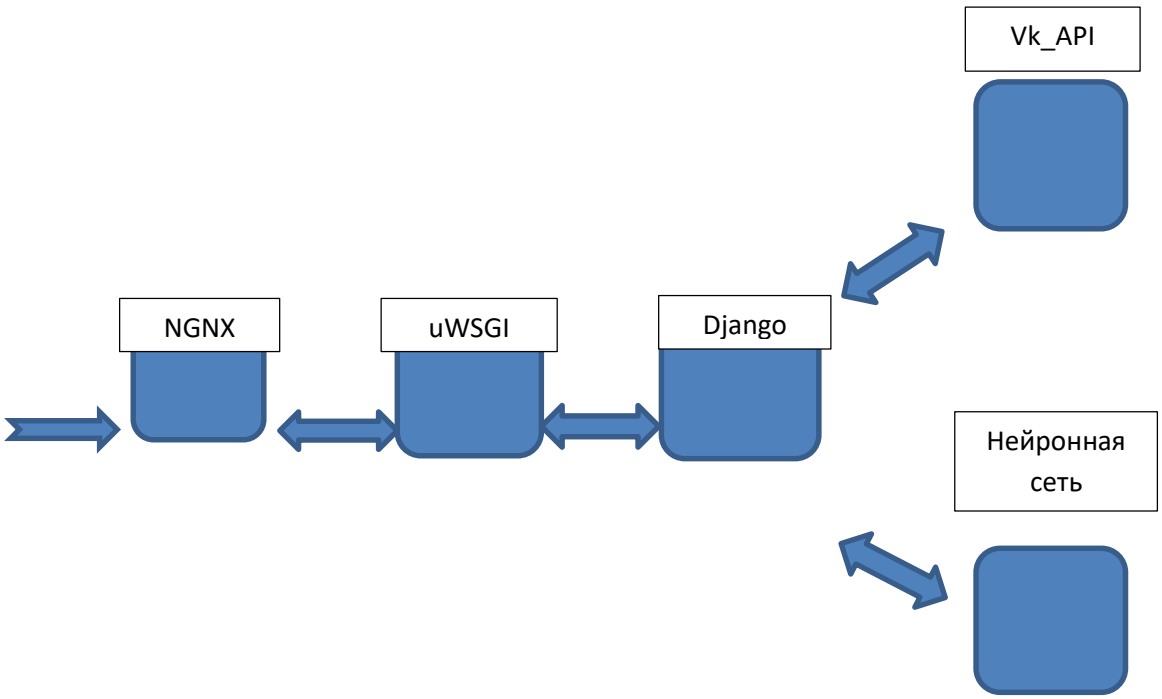
Для взаимодействия с пользователем мы решили сделать удобный сайт, который принимал бы ссылку на пост ВКонтакте и выводил усредненную оценку всех комментариев, а так же примеры комментариев для каждого рейтинга



Архитектура сайта

Сайт построен на взаимодействии веб-сервера и веб-программы. Веб-сервер слушает запрос на определённом порту (в нашем случае порт 8000) и обрабатывает его. Если инструкция для этого запроса существует, то он её выполняет, иначе выдаёт ошибку. В качестве веб-сервера мы используем nginx. Он слушает порт и если на него приходит запрос то он перенаправляет его в веб-программу через интерфейс взаимодействия сервера и программы. В качестве интерфейса мы используем uwsgi (который реализует протокол WSGI). Он предназначен специально для мултипраграмного функционала на python. Поэтому при желании мы можем подключить дополнительные программы. В качестве самой веб-программы используется фреймворк Django, который умеет обрабатывать запросы с веб-сервера и формировать HTML странички. Также к Django подключён шаблонизатор jinja, что позволяет создавать шаблоны страничек и делает архитектуру ещё гибче. В нашем проекте мы не используем базу данных, однако к сайту она уже подключена. Достаточно создать модуль таблицы в Django и мы увидим саму

таблицу в admin-панели сайта. В итоге, удобная модульная архитектура сайта позволяет легко наращивать функционал.



NGINX

Nginx [engine x] — это HTTP-сервер и обратный прокси-сервер, почтовый прокси-сервер, а также TCP/UDP прокси-сервер общего назначения

Мы используем его, как HTTP-сервер, который перенаправляет запросы в веб-приложение. Управление nginx осуществляется с помощью конфигурационных файлов, что очень удобно. Это позволяет модульно подключать новые инструкции к nginx.

uWSGI

uWSGI — веб-сервер и сервер веб-приложений, первоначально реализованный для запуска приложений Python через протокол WSGI

Без WSGI протокола проблематично запустить python, как веб-приложение, потому что голый nginx не способен общаться с Django фреймворком. Поэтому uWSGI, который реализует протокол WSGI, является интерфейсом между Django и nginx. Большим плюсом uWSGI является то, что он способен

вызывать несколько python приложений, которые никак не связаны друг с другом.

Django

Django - это веб-фреймворк Python высокого уровня, который способствует быстрой разработке и чистому, прагматичному дизайну. Созданный опытными разработчиками, он берет на себя большую часть хлопот веб-разработки, поэтому вы можете сосредоточиться на написании своего приложения, не изобретая велосипед. Это бесплатно и с открытым исходным кодом.

Django – ядро сайта. Он обрабатывает запрос и выдаёт HTTP страницу с конечным результатом, используя подключённые к нему модули. Например, нейронная сеть и vk_api. И благодаря модулям можно быстро добавить дополнительный функционал сайту.

В Django встроены:

1. Шаблониатор Jinja – это маленький набор инструкций, который помогает автоматизировать создание HTML шаблонов.
2. Sqlite база данных
3. Режим отладки веб-приложения, что позволяет быстро ловить ошибки

Хостинг

В качестве хостинга мы решили использовать https://my.firstvds.ru/, который может предоставлять различные схемы предустановки операционной системы. Наш сайт работает на ОС Ubuntu-16.04-x86_64 без предустановленного ПО.