ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет Санкт-Петербургская школа
физико-математических и компьютерных наук

Люборт Константин Сергеевич

ОПРЕДЕЛЕНИЕ КАТЕГОРИИ ТОВАРА ДЛЯ ЭЛЕКТРОННОЙ ДОСКИ ОБЪЯВЛЕНИЙ

Выпускная квалификационная работа - БАКАЛАВРСКАЯ РАБОТА
по направлению подготовки *01.03.02 Прикладная математика и информатика*
образовательная программа «Прикладная математика и информатика»

Рецензент
А.С. Найденов

Руководитель
д-р физ.-мат. наук
А.В. Омельченко

Консультант
В.Г. Долбилов

Санкт-Петербург 2021

Оглавление

Автоматическое определение категории товара – одна из важнейших задач, которая встает перед разработчиками сервисов электронной коммерции. Данная задача имеет свои характерные особенности, усложняющие разработку подобных систем: большое количество возможных категорий, плохое качество данных, неравномерное распределение товаров по категориям. Кроме того, решения, предлагающиеся в существующих работах, невозможно сравнить друг между другом в связи с отличающимися датасетами, которые используют участники различных исследовательских групп. В данной работе производится исследование и сравнение различных подходов классификации, которые могут быть применены в данной задаче. На основе наилучшего решения создана система, помогающая пользователям выбрать категорию товара при размещении объявления на одном из крупнейших российских сайтов электронной коммерции. Для адаптации к появлению новых видов товаров разработанное решение регулярно обучается на новых данных. Полученная система полностью интегрирована в работу сайта и протестирована на реальных пользователях.

Keywords: электронная коммерция, классификация товаров, машинное обучение

Automatic item categorization is one of the most important tasks facing the developers of e-commerce services. This task has its own specific features which complicate engineering of such systems: a large number of possible categories, poor data quality, uneven distribution of items between categories. Furthermore, the solutions proposed in the existing works are impossible to compare due to different datasets used by different research groups. In this work, we study and compare different classification approaches that may be applied in this particular task. The best solution served as a basis for a new system meant to help the users choose the category of the product when submitting the advertisement to one of the largest Russian classifieds. In order to adapt to the emergence of new product types the developed solution is regularly trained on new data. The resulting system is fully integrated into the site's workflow and has been tested on real users.

Keywords: e-commerce, product classification, machine learning

Введение

Развитие интернета и веб-технологий каждый год увеличивает популярность сервисов электронной коммерции. Объявления о продаже товаров или предоставлении услуг на таких сервисах чаще всего организованы в иерархическую структуру – дерево категорий. Систематизация контента на сервисе необходима для корректной работы многих аспектов сайта электронной коммерции: она позволяет пользователям более точно находить нужные товары, используется в работе рекомендательных систем, позволяет настраивать дополнительные услуги для отдельных категорий товаров. Также разделение всех объявлений на категории позволяет в автоматическом или ручном режиме указывать атрибуты, применимые только к определенным видам товаров, что позволяет унифицировать отображение объявлений.

При этом, несмотря на выгоды обширного дерева категорий, подробная категоризация всех товаров усложняет подачу объявления для пользователя. Количество возможных категорий в современных сервисах электронной коммерции исчисляется тысячами, что делает сложной задачу выбора подходящей категории при размещении товара или услуги на сайте. При этом данные трудности влияют не только на удовлетворенность продавца при работе с сервисом, но и на качество контента на сайте, что в свою очередь может негативно сказываться на большом число различных продуктовых метрик [10].

Сохранить преимущество категоризации всех товаров на сайте и при этом избавиться от сложностей при размещении объявлений помогают автоматизированные системы, предсказывающие категорию товара при подаче объявления. Задача определения места объявления в таксономии всех товаров формулирует-

ся, как правило, как задача классификации, в которой классами являются все возможные категории товара, а признаками – различные данные относящиеся непосредственно к объявлению или к пользователю, который объявление размещает.

Создание модели классификации, определяющей категорию товара, осложнено характерными для данной задачи проблемами: тысячи классов, высокая зашумленность данных и неравномерное распределение данных по классам. При этом также не существует однозначно верного подхода при решении данной задачи. Это вызвано в первую очередь двумя причинами. Во-первых, исследователи используют различные датасеты в своих работах, что не позволяет сравнить подходы между собой. Во-вторых, некоторые индустриальные решения используют при классификации признаки объявления, специфичные для конкретного сервиса.

Одним из сервисов электронной коммерции, использующим для структурирования контента большое число категорий, является электронная доска объявлений (классифайд) Яндекс.Объявления. Подобно другим классифайдам, продавец при размещении своего товара или услуги на сайте должен указать подходящую категорию для своего объявления. Наличие автоматического определения категории на данном сервисе упростило бы работу продавцов с сайтом.

Цель и задачи

Основная цель данной работы – разработать систему, помогающую пользователям выбрать категорию при размещении объявлений на сервисе Яндекс.Объявления. Данная система должна по введенным пользователем данным определять наиболее возможные категории объявления.

Для этого ставятся следующие задачи:

- Имплементировать и сравнить различные подходы классификации, которые могут быть применены для определения категории объявления. Выбрать наилучшую модель для интеграции в работу сервиса.
- Интегрировать предсказание категории в процесс добавления объявления. Полученная система должна в реальном времени обрабатывать запросы пользователей, используя выбранную модель классификации.
- Протестировать полученное решение на реальных пользователях

Структура работы

В главе 1 представлен обзор существующих работ и индустриальных решений в области классификации товаров.

В главе 2 описаны имплементированные подходы классификации и результаты их offline тестирования.

В главе 3 подробно описана архитектура и технические детали разработанной системы, интегрированной в работу сервиса Яндекс.Объявления и определяющей возможные категории объявлений. Представлен результат опробации решения на реальных пользователях.

1. Обзор литературы

Большую часть подходов, используемых в задаче классификации товаров, можно разделить на два типа. Первый тип решений состоит из одного классификатора, предсказывающего нужную категорию за один шаг исполнения. Такие подходы часто называют "плоскими" (flat-approaches). Второй тип решений использует иерархию классификаторов, каждый из которых отвечает только за предсказание определенного подмножества классов. Предсказание в таких моделях происходит в несколько шагов, где каждый шаг сужает область поиска нужного класса.

Tom Zahavy и др. в своей работе [8] пользуются плоским подходом при классификации товаров. В качестве признаков товара авторы используют заголовок товара и его фотографию. Исследователи сравнивают два различных подхода, которые позволяют совместно использовать данные признаки для определения категории. Первый подход – слияние на уровне признаков (feature-level fusion). При данном подходе каждые из двух признаков обрабатываются отдельно начальными слоями нейронной сети. На определенной глубине выходы начальных слоев конкатенируются, что дает мультимодальное представление товара. Получившееся представления далее классифицируется с помощью нескольких слоев нейронной сети. Второй подход – слияние на уровне решения (decision-level fusion). При таком подходе каждый вход (изображение и название товара) обрабатывается отдельным классификатором, предсказывающим классы решаемой задачи. В дополнение к этому обучается финальная модель, которая совмещает предсказания отдельных классификаторов в итоговый ответ. Наилучшего качества классификации авторам удается добиться с помощью второго подхода. При этом итого-

вая модель превосходит классификатор, использующий только название товара, менее чем на 2% точности (accuracy). Также данная работа позволяет заключить, что является более значимым признаком в задаче классификации товаров: заголовок вносит наибольший вклад в точность полученного решения.

Авторы модели DeepCN [6] в своей работе также используют плоский подход для определения категории товаров в корейском сервисе электронной коммерции Naver Shopping. В своей системе исследователи используют большое количество различных признаков товара для определения его категории. Каждый признак представляется в виде текста и подается на вход рекуррентной нейронной сети (RNN). Выходы отдельных RNN после этого конкатенируются в один итоговый вектор, который классифицируется с помощью нескольких полносвязных слоев нейронной сети. Данная работа является успешным примером того, что использование дополнительных текстовых признаков, кроме заголовка объявления, может улучшить качество модели. Тем не менее большая часть используемых признаков специфична для данного веб-сайта и не может быть использована в общем случае.

Несмотря на то, что плоские подходы могут привлекать простотой в разработке и высокой скоростью работы, использование иерархических подходов способно улучшить качество классификации. Авторы модели HDLTex [7] в своей работе пользуются иерархической зависимостью между классами. Данная работа посвящена не предсказанию категории товаров в сервисах электронной коммерции, но более общей задаче классификации текстов, располагающихся в листьях древовидной иерархической структуры. Тем не менее результаты авторов могут быть применены и к задаче распознавания категории объявлений по следующим причинам: а) текст является главным признаком во всех

работах классификации товаров б) дерево категорий, использующееся в сервисах электронной коммерции, как раз и является способом иерархической организации классов. Исследователи предлагают следующий подход: для каждого узла в дереве категорией тренируется отдельный классификатор, предсказывающий следующий переход вниз по дереву. Итоговый класс находится после нескольких последовательных переходов от корня дерева к листьям. Такой подход позволяет использовать различные архитектуры классификаторов для того, чтобы отделять друг от друга тексты с совершенно различной тематикой (ближе к корню дерева), и для того, чтобы улавливать незначительные отличия между классами (ближе к листьям дерева). С помощью предложенной архитектуры автором удается добиться улучшения в несколько процентов точности (accuracy) в сравнению с плоскими моделями.

Модель KNN-SVM [10] является успешным применением описанной выше идеи иерархической классификации применительно к классификации товаров в в электронной коммерции. Однако, вместо того чтобы использовать создаваемое людьми дерево категорий, авторы предлагают алгоритмически разделять все имеющиеся категории на группы. Алгоритм определения групп похожих категорий устроен следующим образом. Сначала исследователи обучают плоский классификатор, определяющий классы решаемой задачи. Используя данную модель авторы определяют попарные вероятности ошибки между категориями:

$$\mathrm{Conf}(c_1, c_2) = \frac{\sum_{t:f_s(t)\neq f_m(t), f_s(t)\in\{c_1,c_2\}, f_m(t)\in\{c_1,c_2\}}(1)}{\sum_{t:f_s(t)\in\{c_1,c_2\}, f_m(t)\in\{c_1,c_2\}}(1)} \qquad (1)$$

где $f_s(t)$ настоящий класс примера t, а $f_m(t)$ – класс, предсказанный моделью. Далее все классы рассматриваются в качестве

вершин неориентированного графа, где между классами c_1 и c_2 проводится ребром в том случае, если $\mathrm{Conf}(c_1, c_2)$ больше определенного значения. Граф преобразуется в ориентированный с помощью случайного определения направления для всех ребер. В получившемся графе происходит поиск компонент сильной связности, после чего полученные компоненты и являются группами похожих классов. Используя полученные группы, авторы применяют уже описанный подход иерархической классификации: они обучают одну модель (KNN) для определения, к какой группе принадлежит вход, и множество моделей (SVM, одна модель на свое подмножество классов) для определения класса внутри группы. Согласно экспериментам, проведенным авторами, использование алгоритмически генерируемой двухуровневой иерархии улучшает качество классификации по сравнению с подходом, использующим существующее дерево категорий.

Работа Ali Cevahir и Koji Murakami [3] является еще одним примером решения задачи классификации товаров на индустриальных данных. В качестве признаков для классификации исследователи используют название товара и его описание. Авторы также пользуются описанным ранее подходом иерархической классификации. Используемая исследователями иерархия классов состоит из двух уровней и совпадает с деревом категорий сервиса электронной коммерции, который является источником данных в работе. В качестве моделей классификации авторы используют ансамбль из метода ближайших соседей (KNN) и глубоких сетей доверия (DBN). Исследователи делают вывод, что использование описания товара и иерархической классификации повышает качество разработанной модели.

Альтернативный метод использования дерева категорий сервиса электронной коммерции для повышения качества классифи-

кации был предложен [4] исследователями Alibaba Group. Предложенный метод состоит из двух частей. Во-первых, авторы используют отдельные слои внутри нейронной сети, которые определяют принадлежность входа к категориям на промежуточных уровнях дерева категорий. Во-вторых, исследователи определяют функцию потерь, учитывающую иерархическую зависимость между классами. Данный подход отличается от распространенных архитектур для иерархической классификации тем, что авторы используют специальные методы для обучения одной модели, а не обучают иерархию моделей. Согласно экспериментам исследователей, рассматриваемый метод позволяет улучшить точность (accuracy) классификации по сравнению с плоскими моделями, но неизвестно, как данный подход сравнивается с другими иерархическими архитектурами (например, KNN-SVM).

Рассмотренные работы демонстрируют, что заголовок объявления (название товара) является наиболее используемым и эффективным признаком при классификации товаров на сервисах электронной коммерции. При этом использование дополнительных признаков может улучшить качество качество классификации. Также существующие работы показывают, что использование иерархических подходов позволяет увеличить точность (accuracy) решения.

2. Сравнение и выбор модели классификаци

2.1. Данные и используемые метрики

Основную часть датасета для обучения и датасета валидации составляют объявления, которые были размещены на сайте пользователями. При этом около 15% объявлений из-за ошибок пользователей содержат неправильную категорию, что может мешать обучению моделей классификации. Для борьбы с этой проблемой в датасете с большим весом учитываются объявления, в которых категория была исправлена вручную модераторами сервиса. Проведенное в данной работе тестирование показало, что данная модификация датасета улучшает точность (accuracy) классификации финального решения на 0.9%.

Дополнительно в обучающую часть датасета добавляются синонимы названия категории и примеры объявлений, которые указываются в каталоге товаров работниками сервиса Яндекс.Объявления. Использование этих данных при обучении крайне важно для распознавания новых категорий, которые добавляются в каталог товаров и в которые еще не было подано ни одного объявления. Благодаря регулярному обучению модели (глава 3) новая категория будет участвовать в классификации и может быть предложена пользователям сразу же после ее добавления в каталог. При этом необходимо отметить, что использование данных из каталога изменяет распределение количества объявлений в категориях, что может негативно сказаться на качестве классификации. Тем не менее эксперименты с данными, проведенные в данной работе, показали, что добавление этого источника данных ухудшает точность (accuracy) выбранной модели только на

0.15%.

Чтобы исключить пользовательские ошибки в данных, датасет для тестирования состоял из случайно выбранного подмножества объявлений, для которых работниками сервиса Яндекс.Объявления была указана правильная категория. Для удобной разметки в рамках данной работы был создан интерфейс на платформе Яндекс.Толока. Шаг очистки данных для тестирования необходим в связи с тем, что ошибки пользователей могут быть предвзяты в сторону неправильных предсказаний предварительных моделей классификации, которые были разработаны в данной работе и использовались на начальных этапах работы сервиса Яндекс.Объявления. Таким образом, сравнение различных подходов классификации на зашумленных данных может привести к тому, что большую точность получат модели, которые и сгенерировали неправильные предсказания, выбранные пользователями.

Итоговый датасет состоит из 800 тысяч примеров для обучения, 200 тысяч примеров для валидации и 20 тысяч примеров для тестирования. В датасете около 2400 категорий.

В качестве основных метрик для сравнения используется accuracy и top-k accuracy. Top-k accuracy в общем виде формулируется как доля тестируемых примеров, для которых правильная категория принадлежала k категориям с наибольшей предсказанной вероятностью. Использование top-k accuracy мотивируется тем, что при заполнении объявления должны быть показаны несколько наиболее вероятных категорий, из которых пользователю необходимо выбрать подходящую.

2.2. Плоские модели

При разработке моделей машинного обучения желательно иметь baseline – простое в разработке и широко используемое решение, с которым можно будет в дальнейшем сравнить более сложные подходы. В качестве такого решения в данной работе был выбран Наивный байесовский классификатор, использующийся для сравнения во многих статьях об обработке естественного языка [12].

Было протестировано три версии Наивного байесовского классификатора, популярных в задаче классификации текстов: мультивариативный подход, мультиномиальный подход, а также комплементарная версия классификатора, являющаяся улучшением мультиномиального подхода и представленная Rennie и др [11]. При векторизации были рассмотрены варианты использования только униграмм (отдельных слов) и совместного использования униграмм и биграмм (последовательности из двух идущих подряд слов). Для мультиномиального и комплементарного подхода тестировалась векторизация на основе числа вхождения N-граммы в документ и TF-IDF. Также была протестирована лемматизация при предобработке текста с использованием библиотеки MyStem. Все гиперпараметры (наличие лемматизации, длины N-грамм при векторизации, тип векторизации, параметр сглаживания Лапласа) подбирались на валидационном датасете с помощью поиска по сетке. Наилучшие результаты для каждой из трех версий Наивного байесовского классификатора представлены в таблице 1

Модель	Accuracy	Top-3 Acc.	Top-5 Acc.
Мультивариативный NB униграммы + биграммы Лемматизация: нет	67.4	83.4	87.9
Мультиномиальный NB униграммы + биграммы Векторизация: TF-IDF Лемматизация: нет	**67.6**	**84.0**	**88.8**
Комплементарный NB униграммы + биграммы Векторизация: TF-IDF Лемматизация: да	65.7	83.0	88.1

Таблица 1: результаты тестирования Наивного байесовского классификатора

Одни из лучших результатов в задачах обработки естественного языка показывают модели, основанные на архитектуре Transformer [1]. В связи с этим было решено в качестве одного из возможных решений протестировать качество классификации с использованием предобученной модели DistilBERT [5]. Выбор модели меньшего размера в сравнении с оригинальной моделью BERT [2] обусловлен ее более быстрым исполнением, что является важной характеристикой для дальнейшего использования решения в реальных условиях. Кроме того, на данных, используемых в этой работе, использование DistilBERT вместо BERT приводило к потере всего 0.3% accuracy.

Модель дообучалась без заморозки весов с маленькой скоростью обучения, что соотносится с рекомендуемым [9] способом дообучения моделей основанных на Transformer. Гиперпараметры модели (конфигурация слоев после базовой нейронной сети,

скорость обучения и размер батча) подбирались на валидационном датасете. Результат тестирования наилучшей конфигурации представлен в таблице 2

Модель	Accuracy	Top-3 Acc.	Top-5 Acc.
Pretrained DistilBERT	71.8	89.3	93.6

Таблица 2: результаты дообучения DistilBERT

2.3. Иерархические модели

Следующим решением, имплементированным и протестированным в данной работе, является подход KNN-SVM, хорошо показавший себя в другом сервисе электронной коммерции [10]. При этом тестирование данного решения в оригинальном виде дало достаточно маленький прирост к accuracy в сравнении с одношаговой классификацией только с помощью KNN (таблица 3). Можно гипотетически предположить, что алгоритм определения групп похожих категорий, предложенный авторами, может работать значительно хуже для менее подробных деревьев категорий: дерево категорий сервиса Ebay содержит 20 тысяч листьевых категорий, когда как в данной работе рассматривается дерево с 2.4 тысячами листьев. В том числе данный алгоритм не может определить группы, состоящие только из 2 категорий.

В данной работе предлагается улучшение данного алгоритма, показавшее более высокую точность (accuracy) классификации на используемом датасете. Используя вероятности ошибки между категориями ($\mathrm{Conf}(c_1, c_2)$) из оригинальной статьи (формула 1), определим попарные расстояния между категориями следующим образом:

$$\text{Dist}(c_1, c_2) = \begin{cases} 1 - \text{Conf}(c_1, c_2) & \text{if } c_1 \neq c_2 \\ 0 & \text{if } c_1 = c_2 \end{cases}$$

Тогда вместо графового алгоритма, который использовался авторами статьи, можно воспользоваться алгоритмами кластеризации, не требующими выполнения неравенства треугольника. Получившиеся кластеры в таком случае и будут группами похожих категорий. Для имеющегося датасета в данной работе наилучшие результаты показала иерархическая кластеризация с методом средней связи. Сравнение оригинальной версии KNN-SVM и версии с использованием предложенной модификации представлено в таблице 3.

Существуют и другие варианты улучшения рассматриваемого подхода. Например, можно заменить модель на одном из уровней классификации на более мощную. В данной работе было протестировано решение с заменой первого шага классификации на DistilBERT. Accuracy получившейся модели выше (таблица 3), чем в оригинальном варианте с KNN, но все равно остается более низкой в сравнении с одношаговой классификацией с помощью использованной нейронной сети.

Модель	Accuracy	Top-3 Acc.	Top-5 Acc.
KNN	67.5	84.2	88.3
KNN-SVM	67.8	84.5	88.6
KNN-SVM +Кластеризация	69.1	85.2	89.5
DistilBERT + SVM +Кластеризация	**71.5**	**87.7**	**92.0**

Таблица 3: результаты тестирования KNN-SVM

Также в данной работе был имплементирован и протестирован альтернативный подход к иерархической классификации, предложенный Alibaba Group. [4]. Как было упомянуто ранее, данный подход заключается в использовании дополнительных слоев нейронной сети и функции потерь, учитывающих иерархию категорий. В качестве базовой нейронной сети была использована нейронная сеть DistilBERT, показавшая хорошие результаты в предыдущих экспериментах. Подбор гиперпараметров и тестирование проводились в двух вариантах: с использованием иерархии категорий из продуктового каталога сервиса Яндекс.Объявления и с использованием двухуровневой иерархии из подхода KNN-SVM с предложенной модификацией (кластеризация). Несмотря на незначительное улучшение точности (accuracy), данный подход проигрывает обычному дообучению DistilBERT в top-3 и top-5 accuracy. Результаты тестирования представлены в таблице 4.

Модель	Accuracy	Top-3 Acc.	Top-5 Acc.
DistilBERT + Alibaba Кластеризация	71.7	89.0	93.3
DistilBERT + Alibaba Каталог	71.9	88.7	93.0

Таблица 4: результаты тестирования подхода Alibaba Group

2.4. Дополнительные признаки

Также в данной работе было протестировано использование описания объявления в качестве дополнительного признака совместно с заголовком объявления. Для этого производилась конкатенация заголовка и описания, на получившихся текстах до-

обучалась языковая модель DistilBERT. При этом в связи с тем, что описания часто имеют большую длину, многократно возрастает скорость исполнения модели. Для борьбы с этим длина токенизированного текста искусственно обрезалась до 128 токенов (приблизительно 50 первых слов). Было выявлено, что данный дополнительный признак улучшает качество классификации (таблица 5). Тем не менее, даже с учетом искусственного уменьшения длины текста, модель работает в среднем на 70% дольше (110ms при использовании описания, 65ms при использовании только заголовка объявления). В связи с тем, что модель должна использоваться для обработки запросов в реальном времени, данный подход не рассматривался при выборе финального решения.

Модель	Accuracy	Top-3 Acc.	Top-5 Acc.
DistilBERT + Описание	73.1	89.3	93.4

Таблица 5: Использование описание как дополнительного признака

2.5. Выбор модели

Далее представлена сводная таблица с результатом тестирования всех решений, использующих только заголовок объявления (таблица 6).

Модель	Accuracy	Top-3 Acc.	Top-5 Acc.
Мультиномиальный NB	67.6	84.0	88.8
Pretrained DistilBERT	71.8	**89.3**	**93.6**
KNN-SVM	67.8	84.5	88.6
KNN-SVM +Кластеризация	69.1	85.2	89.5
DistilBERT + SVM +Кластеризация	71.5	87.7	92.0
DistilBERT + Alibaba Кластеризация	71.7	89.0	93.3
DistilBERT + Alibaba Каталог	**71.9**	88.7	93.0

Таблица 6: Все решения, использующие заголовок объявления

Для дальнейшей интеграции в сервис Яндекс.Объявления было выбрано дообучение сети DistilBERT. Это связано с тем, что данное решение показало наилучшие результаты по top-3 accuracy и top-5 accuracy и лишь незначительно отстает по обычной точности (accuracy) от модификации из решения Alibaba Group.

2.6. Выводы и результаты по главе

В процессе выбора модели классификации было протестировано 8 различных подходов. Гиперпараметры всех рассматриваемых моделей классификации подбирались с использованием валидационного датасета. Также были проведены различные эксперименты с данными, использующимися для обучения. В том числе с помощью учета одной из частей датасета с большим весом получилось улучшить точность (accuracy) классификации на 0.9%. Также улучшение точности классификации показало ис-

пользование описания объявления в качестве дополнительного признака. Использование описания не было включено в финальное решение в связи с большим временем работы модели.

Наилучшее соотношения качества классификации и времени исполнения показало решение, основывающееся на дообучении нейронной сети DistilBERT, которое было выбрано в для интеграции в сервис Яндекс.Объявления.

3. Разработанная система

В данной главе описывается интеграция выбранной модели в работу сервиса Яндекс.Объявления. Рассмотрим архитектуру финального решения в целом, а затем подробно рассмотрим некоторые его части.

На схеме 1 представлена архитектура итоговой системы. Красным цветом выделены компоненты системы, которые были полностью разработаны и реализованы в рамках данной работы. Зеленым цветом отмечены компоненты, существовавшие до реализации системы, но в которые были внесены изменения.

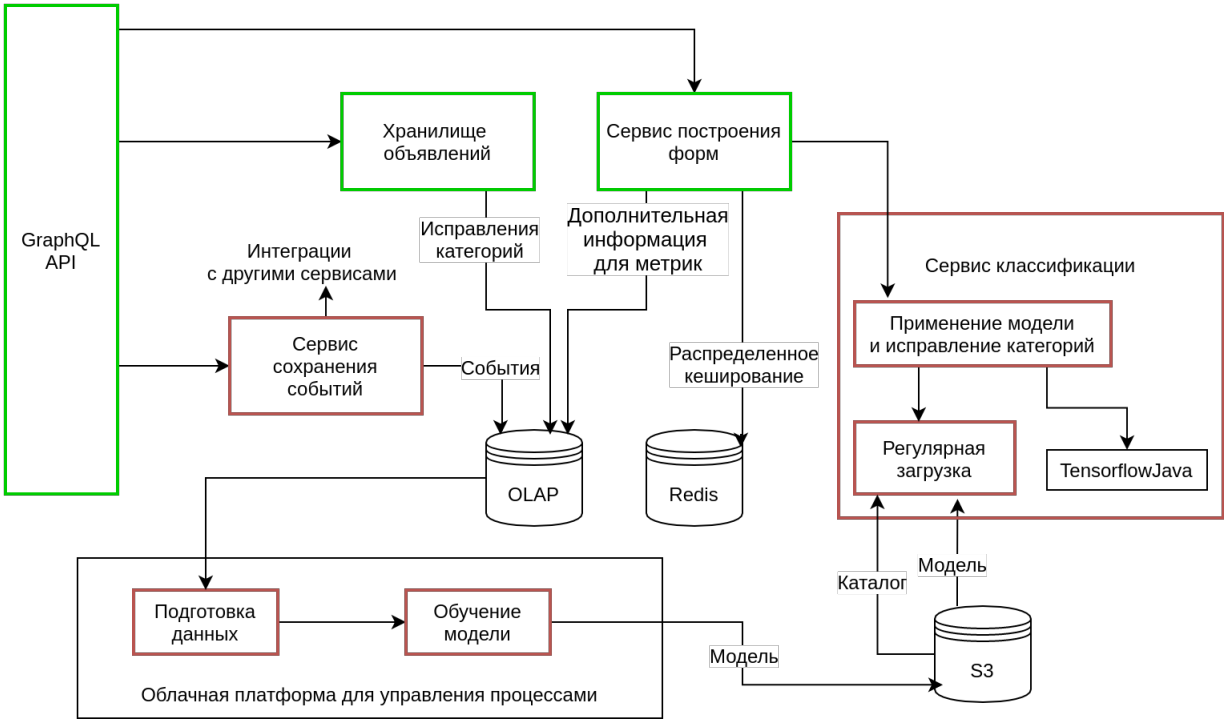

Рис. 1: Архитектура системы

3.1. Регулярное обучение модели

Один из вариантов интеграции модели классификации в сервис – однократно обучить модель на имеющихся данных и со-

здать систему, которая бы работала только с этой версией модели. Тем не менее такой подход имеет несколько принципиальных минусов. Во-первых, в мире регулярно появляются новые виды и бренды товаров. В связи с этим качество работы системы будет со временем падать, так как однократно обученная модель классификации не сможет распознавать некоторые продаваемые товары, которых не существовало в момент обучения. Во-вторых, структура каталога сервиса Яндекс.Объявления постоянно меняется: добавляются новые категории; существовавшие категории разделяются на несколько категорий или наоборот объединяются в одну. В связи с этим было принято решение сделать обучение модели регулярным процессом. Один раз в 3 дня обучается новая модель классификации, которая при обучении использует новые объявления и актуальную версию каталога.

Для регулярного запуска процесса обучения было принято решение использовать платформу для управления вычислительными процессами "Нирвана", а не создавать отдельный микросервис. Весь процесс регулярного обучения модели состоит из нескольких этапов: загрузка данных и создание обучающего и валидационного датасета, обучение новой версии модели, загрузка модели в объектное хранилище.

При создании датасета для обучения и валидации используются данные из трех источников, указанных в главе 2: поданные пользователями объявления, исправления категорий модерацией сервиса и данные из каталога товаров. Для возможности использования исправления категорий было необходимо внести доработки в микросервис хранилища объявлений. В ходе данной работы было реализовано сохранение информации об исправлениях категории в аналитическую базу данных, из которой производится загрузка данных в процессе регулярного обучения.

При этом даже использование дополнительных данных из каталога товаров не гарантирует отсутствия категорий с недостаточным числом примеров для обучения (например, если при создании новой категории примеры товаров не были указаны). В связи с этим был реализован следующий подход: категории, в которых число примеров менее 10 и которые имеют общего непосредственного родителя в дереве категорий, объединяются в одну категорию (в их непосредственного родителя). При размещении объявления пользователь самостоятельно уточняет предсказание классификатора, выбрав подходящего ребенка предложенной нелистьевой категории.

После создания датасета описанным выше образом, происходит обучение модели, выбор которой описан в главе 2. Обучение модели происходит с использованием библиотеки Tensorflow. Далее обученная модель, отображение выходов модели в выбранные категории каталога, а также данные для прогрева модели сохраняются в объектное хранилище. Для возможности восстановления при обнаружении каких-либо проблем неактуальные версии модели хранятся в течение 30 дней, после чего удаляются из хранилища.

3.2. Применение модели и интеграция с формой добавления объявлений

Следующая задача, которую необходимо было решить в данной работе – как именно обученные модели будут запускаться для классификации объявлений в реальном времени. Одним из вариантов применения Tensorflow моделей является использование системы Tensorflow Serving. Данная система представляет собой готовый к запуску на своем оборудовании сервис, предо-

ставляющий API для запуска моделей классификации и поддерживающий регулярную загрузку моделей из хранилища. Но использование Tensorflow Serving при интеграции с сервисом Яндекс.Объявления осложнено тем, что результаты применения модели должны быть обработаны в соответствии с текущим состоянием продуктового каталога сервиса. Например, в некоторых сценариях подачи объявления заранее известно, что необходимая пользователю категория является ребенком категории "Транспорт и запчасти", а потому не стоит возвращать пользователю не подходящие под данное условие категории. Интеграция данной логики в рассматриваемую систему трудоемка, так как имеющиеся вспомогательные модули для работы с каталогом написаны на языке Scala, а Tensorflow Serving реализован на языке C++.

Альтернативное решение, которое было выбрано в данной работе – реализация отдельного микросервиса, который запускает обученные модели с помощью библиотек, поддерживающих запуск моделей из JVM языков. В качестве такой библиотеки были выбраны официальные JVM-биндинги для Tensorflow. Микросервис разработан на языке Scala и предоставляет gRPC API.

Кроме непосредственно запуска моделей, в разработанном микросервисе также происходит актуализация предсказаний моделей в соответствии с текущим состоянием продуктового каталога. Как было упомянуто ранее, каталог сервиса динамический, а потому категории обученной модели могут несоответствовать текущему состоянию каталога. Тем не менее некоторые незначительные несоотвтетсвия могут исправлены в момент применения модели без ее обучения на новых данных. Например, если несколько категорий из множества S сливаются в одну категорию C, то S необходимо удалить из результатов, а в качестве вероятности C можно использовать сумму вероятностей катего-

рий из S.

Также при разработке указанного микросервиса необходимо было решить задачу регулярной загрузки новых версий продуктового каталога и обученной модели. При этом загрузка обученной модели представляет собой простой случай решаемой задачи, так как использование модели происходит только в одном месте программного кода. В таком случае достаточно отдельного потока, скачивающего модель с определенным промежутком времени. Сложности же возникают, если несколько различных частей программы требуют регулярного обновления одного и того же ресурса (каталог – именно такой случай). В данном случае неэффективно загружать один и тот же ресурс несколько раз. Для решения данной задачи был разработан отдельный компонент для регулярного обновления ресурсов, основывающийся на шаблоне проектирования "Издатель-Подписчик". Разработанный компонент позволяет подписаться на определенный ресурс и получать его обновления с промежутком времени, который задается конфигурацией ресурса. Если ресурс требуется многим подписчикам, то на каждое обновление он будет загружен из сети фоновым потоком только один раз. Если же у ресурса не остается подписчиков, то ресурс перестает загружаться. Также в данном компоненте имплементировано кэширование последней версии загруженного ресурса. Это позволяет подписчику получить ресурс сразу, не дожидаясь его следующего обновления. Разработанный компонент успешно используется и в других частях сервиса Яндекс.Объявления.

Имплементированный микросервис было необходимо интегрировать с другими частями системы Яндекс.Объявления. В данной работе была выполнена вся необходимая интеграция с бэкендом сервиса. До выполнения данной работы процесс размещения

объявления выглядел следующим образом: пользователь выбирал необходимую ему категорию, а затем попадал на страницу с самой формой подачи объявления. Так как модель классификации должна использовать данные самого объявления, то необходимо было сделать выбор категории отдельным шагом на форме. В таком случае пользователь должен сначала ввести часть данных своего объявления, а потом выбрать одну категорию из предложенных моделью. Формирование формы (содержание и порядок шагов) происходит в отдельном микросервисе. Добавление нового шага потребовало нескольких доработок микросервиса построения форм, описанных далее.

Во-первых, предыдущая версия рассматриваемого микросервиса не позволяла сформировать форму частично (сформировать только несколько первых шагов). При этом некоторые шаги формы, например, атрибуты товара, зависят от категории. Таким образом, пока пользователем не пройден новый шаг выбора категории из списка, некоторые последующие шаги построить невозможно. Для решения данной проблемы была добавлена возможность специфицировать для каждого шага набор зависимостей – данных, которые должны быть заполнены пользователем для построения данного шага. Если зависимости не выполнены, то данный шаг и все последующие не формируются.

Во-вторых, микросервис построения форм до выполнения данной работы не имел возможности сохранения состояния в базе данных. При каждом изменении данных на форме происходил запрос к рассматриваемому микросервису, который вычислял порядок и содержание всех шагов заново. Данная особенность привела бы к тому, что при заполнении любого шага был бы совершен запрос к разработанному сервису классификации, даже если заголовок объявления не был изменен. Классификация

объявления – ресурсоемкая операция, которая может занимать до нескольких сотен миллисекунд. Для устранения данной задержки была разработана возможность распределенного кэширования шагов формы. Кэширование при этом обязано быть именно распределенным, так микросервис построения форм и микросервис классификации запускаются во множественных экземплярах и отсутствует гарантия того, что различные запросы одного пользователя будут обработаны одними и теми же экземплярами микросервисов. В качестве базы данных было решено использовать Redis. При этом существовавшие библиотеки для работы с Redis, совместимые с библиотекой для асинхронного программирования ZIO, которая используется в проекте Яндекс.Объявления, не поддерживали режим кластера данной БД. В связи с этим был разработан интерфейс для использования Redis вместе с ZIO и соответствующий адаптер для библиотеки Jedis. Имплементированное решение экспортирует метрики времени ответа и числа ошибок при работе с БД. Функциональность адаптера для Jedis была покрыта модульными тестами. Разработанный клиентский модуль для работы с Redis успешно переиспользуется в других компонентах сервиса Яндекс.Объявления. Поддержка распределенного кэширование шагов формы позволила снизить нагрузку (количество запросов в единицу времени) на микросервис классификации в 5 раз.

3.3. Сбор пользовательских метрик и результаты онлайн-тестирования

Кроме проведенного offline тестирования моделей классификации необходимо было провести online тестирование разработанной системы: апробацию на реальных пользователях. В ка-

честве метрик для такого тестирования было выбрано два показателя:

- Как часто пользователи выбирают категорию из нескольких категорий, предложенных системой, а не указывают категорию вручную с помощью навигации по продуктовому каталогу.
- Для какого процента объявлений пользователями была выбрана категория, которую модель классификации считает наиболее вероятной.

Для сбора указанных метрик можно было бы воспользоваться готовыми системами для веб-аналитики, например, системой Яндекс.Метрика. Тем не менее, несмотря на простоту интеграции, использование такой системы не позволяет собирать специфичные для каждого процесса размещения данные. Например, для дальнейшего улучшения классификации хотелось бы понимать не только долю случаев, когда предложенные категории не подходят пользователю, но и в каких конкретных категориях модель ошибается чаще всего.

В связи с указанным ограничением было принято решение собирать данные для аналитики с помощью логирования всех пользовательских действий на сайте. Была спроектирована и разработана модель для всех основных действий пользователей – "событий". Разработанная модель включает в себя не только сам факт наступления события (например, выбора категории), но и специфичные для каждого события данные (например, какие еще категории были показаны пользователю).

Для сохранения событий был спроектирован и имплементирован отдельный микросервис с gRPC API. Информация о пользовательских действиях отправляется фронтендом сервиса Ян-

декс.Объявления, имплементированный микросервис в зависимости от типа события запрашивает из других компонентов сайта дополнительную информацию, необходимую для дальнейшей аналитики, и производит сохранение данных в аналитическую базу данных. Имплементированное решение используется не только для сбора метрик классификации объявлений, но и для аналитики других аспектов работы электронной доски объявлений.

С помощью реализованного подхода были собраны пользовательские метрики процесса классификации объявлений:

- Для 95.3 % объявлений пользователи выбирают одну из категорий, предложенных моделью, а не пользуются ручным выбором категории.
- Для 70.9 % объявлений пользователями выбирается первая категория, предложенная моделью классификации.

3.4. Выводы и результаты по главе

Процесс классификации объявлений при их размещении был полностью поддержан на бэкенд стороне сайта Яндекс.Объявления. Раз в два дня происходит обучение модели на новых собираемых данных. Был имплементирован микросервис классификации объявлений, принимающий данные объявления и возвращающий наиболее вероятные категории. Работа с микросервисом классификации поддержана в микросервисе построения формы добавления, который также была добавлена поддержка распределенного кэширования результатов классификации. Поддержка кэширование позволила снизить нагрузку на микросервис классификации в 5 раз.

Полученное решение протестировано на реальных пользователях сервиса. В 70.9 % случаев пользователям подходит первая

предложенная категория. Для 95.3 % объявлений пользователи пользуются разработанной системой, а не выбирают категорию вручную с помощью навигации по продуктовому каталогу.

Заключение

Основным результатом данной работы является создание системы для классификации объявлений и ее интеграция в работу сервиса Яндекс.Объявлений. В ходе данной работы были достигнуты следующие основные результаты:

- Были обучены и протестированы различные модели классификации, применимые к задаче определения категории объявлений. В том числе производилось тестирование существующих индустриальных решений. Для дальнейшей интеграции была выбрана модель, показавшая 71.8% точности (accuracy) классификации и удовлетворяющая требованиям производительности.

- Имплементирован регулярный сбор актуальных данных и обучение новой версии выбранной модели, что позволяет решению адаптироваться под новые виды товаров и новые категории в продуктовом каталоге сервиса. Новая версия модели классификации загружается на сервис один раз в 3 дня.

- Разработан микросервис классификации, который, используя выбранную модель, определяет категорию объявления при его размещении на сайте. Была проведена вся необходимая интеграция микросервиса с бэкендом сервиса Яндекс.Объявления.

- Поддержан сбор пользовательских метрик и полученная система была протестирована на реальных пользователях сервиса. В 70.9 % добавлений нового объявления пользователям подходит первая категория, предлагаемая системой. В 95.3 % случаев пользователи выбирают категорию

не вручную, а пользуются предложениями разработанной системы.

Дальнейшая работа возможна в следующих направлениях:

- Существуют и другие признаки объявления, которые можно использовать при классификации. В том числе предполагается, что в дальнейшем можно улучшить качество классификации при использовании фотографии товара.
- Использование описания объявления в качестве дополнительного признака позволило улучшить точность (accuracy) решения, но не было учтено в итоговой модели в связи со значительным увеличением времени ее работы. В качестве дальнейшей работы можно оптимизировать скорость работы системы, что позволит использовать этот признак для обработки запросов в режиме реального времени.

Список литературы

[1] Attention is all you need / Ashish Vaswani, Noam Shazeer, Niki Parmar et al. // Advances in Neural Information Processing Systems. — 2017. — P. 6000—6010.

[2] Bert: Pre-training of deep bidirectional transformers for language understanding / Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova // arXiv preprint arXiv:1810.04805. — 2018.

[3] Cevahir Ali, Murakami Koji. Large-scale Multi-class and Hierarchical Product Categorization for an E-commerce Giant // Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. — 2016. — P. 525–535.

[4] Deep Hierarchical Classification for Category Prediction in E-commerce System / Dehong Gao, Wenjing Yang, Huiling Zhou et al. // arXiv preprint arXiv:2005.06692. — 2020.

[5] DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter / Victor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf // arXiv preprint arXiv:1910.01108. — 2019.

[6] Ha Jung-Woo, Pyo Hyuna, Kim Jeonghee. Large-scale item categorization in e-commerce using multiple recurrent neural networks // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. — 2016. — P. 107–115.

[7] Hdltex: Hierarchical deep learning for text classification / Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa

et al. // 2017 16th IEEE international conference on machine learning and applications (ICMLA) / IEEE. — 2017. — P. 364–371.

[8] Is a picture worth a thousand words? A deep multi-modal architecture for product classification in e-commerce / Tom Zahavy, Abhinandan Krishnan, Alessandro Magnani, Shie Mannor // Proceedings of the AAAI Conference on Artificial Intelligence. — Vol. 32. — 2018.

[9] Peters Matthew E, Ruder Sebastian, Smith Noah A. To tune or not to tune? adapting pretrained representations to diverse tasks // arXiv preprint arXiv:1903.05987. — 2019.

[10] Shen Dan, Ruvini Jean-David, Sarwar Badrul. Large-scale item categorization for e-commerce // Proceedings of the 21st ACM international conference on Information and knowledge management. — 2012. — P. 595–604.

[11] Tackling the poor assumptions of naive bayes text classifiers / Jason D Rennie, Lawrence Shih, Jaime Teevan, David R Karger // Proceedings of the 20th international conference on machine learning (ICML-03). — 2003. — P. 616–623.

[12] Text classification algorithms: A survey / Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa et al. // Information. — 2019. — Vol. 10, no. 4. — P. 150.