МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

РЕШЕНИЕ ЗАДАЧИ ИНФОРМАЦИОННОГО ПОИСКА С ПРИМЕНЕНИЕМ DEEP STRUCTURED SEMANTIC MODEL И ТЕХНОЛОГИИ CATBOOST

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 451 группы
направления 09.03.04 — Программная инженерия
факультета КНиИТ
Нарцева Андрея Дмитриевича

Научный руководитель
зав. каф. техн. прогр,
к. ф.-м. н., доцент _____ И. А. Батраева

Заведующий кафедрой
к. ф.-м. н., доцент _____ А. С. Иванов

Саратов 2020

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3
1 Применение Deep Structured Semantic Model и CatBoost для решения задачи информационного поиска 5
1.1 Задача информационного поиска 5
1.2 Deep Structured Semantic Model 6
1.3 Ранжирование поисковой выдачи 8
1.4 Предложенный метод решения задачи 9
2 Реализация и обучение моделей 10
2.1 Предварительная обработка данных 10
2.2 Реализация и обучение DSSM для решения задачи поиска 11
2.3 Обучение модели CatBoost для решения задачи ранжирования 12
2.4 Анализ полученных результатов 13
ЗАКЛЮЧЕНИЕ 14
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 15

ВВЕДЕНИЕ

Информационный поиск родился на стыке библиотечного дела и информатики в середине XX века. Долгое время он оставался скромной дисциплиной, которой занималось небольшое количество исследователей. Несмотря на то, что информационный поиск возник задолго до появления интернета, именно веб-поиск задает основные тенденции развития данной области [1]. Перед современными поисковыми системами ставится задача ранжирования документов на основе их смыслового соответствия запросу, а не просто по совпадению ключевых слов. Переход на семантический уровень поиска не может быть осуществлен только на основе имеющихся классических алгоритмов: на помощь приходят методы машинного обучения.

Задача информационного поиска на практике разбивается на две подзадачи: поиск релевантных документов и ранжирование результатов поиска. Большой размер коллекции документов и строгие временные ограничения делают невозможным применение сложных моделей ко всем документам коллекции [2]. Поэтому обычно поисковая выдача формируется поэтапно: применение простых и быстрых методов позволяет отобрать кандидатов для последующего ранжирования более сложными, но медленными моделями.

Для решения задачи поиска хорошо зарекомендовали себя глубокие нейронные сети, осуществляющие построение метрических пространств, в которых расстояния между векторами соответствуют семантическим отношениям между исходными объектами. К таким моделям относится Deep Structured Semantic Model (DSSM), предложенная исследователями из компании «Microsoft». В работе [3] отмечается, что DSSM показывает лучшее качество среди моделей данного класса. DSSM применяется во многих поисковых системах, в том числе и в поисковом алгоритме «Палех», разработанном в компании «Яндекс» [4]. Однако, несмотря на широкую область применения DSSM, open-source реализации модели на текущей момент нет. Поэтому создание реализации, позволяющей проводить эксперименты с архитектурой и тонкую настройку гиперпараметров, представляет интерес.

В современных поисковых системах ранжирование поисковой выдачи происходит на основе большого числа факторов различной природы. Поэтому для решения данной задачи обычно используется градиентный бустинг: этот метод машинного обучения хорошо работает с данными, структура которых

неоднородна. В данной работе для реализации ранжирующей модели была выбрана технология CatBoost. Согласно результатам сравнения [5], CatBoost показывает наилучшее качество по сравнению с открытыми аналогами.

Целью данной работы является решение задачи информационного поиска с применением Deep Structured Semantic Model и технологии CatBoost.

Для выполнения поставленной цели необходимо:

- выполнить предварительную подготовку данных для приведения их к виду, пригодному для дальнейшего анализа;
- решить проблему отсутствия в датасете отрицательных примеров;
- реализовать Deep Structured Semantic Model и необходимые для ее обучения и дальнейшего использования инструменты;
- провести эксперименты по обучению Deep Structured Semantic Model для решения задачи поиска;
- осуществить подбор гиперпараметров и обучить модель CatBoost для решения задачи ранжирования;
- замерить метрики качества поиска и проанализировать полученные результаты.

Бакалаврская работа состоит из раздела «Обозначения и сокращения», введения, двух разделов основной части, заключения, списка использованных источников и четырех приложений. Общий объем работы — 74 страницы, из них 61 страница — основное содержание, включая 8 рисунков и 2 таблицы. В списке использованных источников содержится 51 наименование.

Первый раздел «Применение Deep Structured Semantic Model и CatBoost для решения задачи информационного поиска» содержит описание используемых моделей машинного обучения и все необходимые предварительные сведения для практической реализации предложенного метода решения задачи.

Во втором разделе «Реализация и обучение моделей» рассмотрена реализация используемых моделей, описаны проведенные эксперименты по обучению, приведены результаты замера метрик качества поиска.

1 Применение Deep Structured Semantic Model и CatBoost для решения задачи информационного поиска

1.1 Задача информационного поиска

Информационный поиск (ИП) — это процесс поиска в большой коллекции неструктурированного материала, удовлетворяющего информационные потребности [1]. Под «неструктурированными» подразумеваются данные, не имеющие строгой очевидной структуры. Однако к информационному поиску нередко относят задачи, которые связаны с работой с «полуструктурированными» данными, имеющими скрытую структуру (например, тексты на естественных языках).

Классическая задача информационного поиска — дана коллекция документов и множество поисковых запросов, требуется для каждого запроса предоставить список наиболее релевантных ему документов из этой коллекции. *Релевантностью* называют степень соответствия документа заданному запросу.

В поисковых системах список документов, являющихся результатом обработки пользовательского запроса, называется *поисковой выдачей*. Как правило, документы в выдаче упорядочены по мере уменьшения некоторой метрики, являющейся мерой релевантности документа запросу, который породил данную выдачу. Этот процесс называют *ранжированием поисковой выдачи*. Таким образом, сформулированная задача информационного поиска на практике разбивается на две подзадачи: поиск релевантных документов и ранжирование результатов поиска.

В настоящее время основным направлением развития информационного поиска является веб-поиск, с которым связаны последние достижения в данной области. В силу вышесказанного, будем рассматривать информационный поиск в контексте веб-поиска, не затрагивая при этом технические аспекты, которые иногда относят к информационному поиску (например, сбор информации и формирование базы документов). В данной работе задача информационного поиска рассматривается в контексте *классической задачи информационного поиска* и заключается в поиске документов, удовлетворяющих запросу, в некоторой коллекции документов. Специфика веб-поиска заключается в том, что коллекция документов имеет значительный размер, а к реализуемому решению предъявляются строгие требования по быстродействию.

1.2 Deep Structured Semantic Model

В 2013 году исследователи из компании «Microsoft» в работе [3] предложили свой подход к решению задачи веб-поиска, который получил название Deep Structured Semantic Model (DSSM). Модель имеет широкую область применения: начиная с информационного поиска и заканчивая рекомендательными системами. В частности, DSSM используется в поисковом алгоритме «Палех», разработанном в компании «Яндекс» [4].

В основе Deep Structured Semantic Model лежат две идеи: обучение латентной семантической модели с учителем, а также применение глубоких нейронных сетей для семантического моделирования.

На вход модели подаются векторизованные представления запросов и документов. Как правило, это вектор высокой размерности, в котором для каждого выделенного терма указано количество раз, сколько этот терм встречается в исходном тексте. Ко входам модели применяются преобразования, заданные скрытыми слоями, осуществляющие отображение запросов и документов в единое семантическое векторное пространство, размерность которого невысока (обычно несколько сотен). Релевантность документа запросу вычисляется как близость между полученными семантическими векторами.

Deep Structured Semantic Model может использоваться для решения задач классификации, мультиклассификации, регрессии и ранжирования [6]. Широкий круг задач определяет разнообразие используемых функций потерь — поддерживается обучение в поточечном и попарном режимах:

1. *Точечные функции потерь*. При поточечном обучении сети на вход DSSM подаются пары «запрос — документ», для которых вычисляется выход сети, к которому и применяется выбранная функция потерь. При таком подходе наиболее часто используются: бинарная кросс-энтропия, категориальная кросс-энтропия, средняя квадратичная ошибка.
2. *Попарные функции потерь*. Попарное обучение заключается в том, что на вход сети подаются тройки $(q, d+, d-)$, где q — некоторый запрос, $d+$ — релевантный ему документ, а $d-$ — нерелевантный документ. К данному виду функций потерь относится *Triplet-loss*, описанная в работе [7]. Сеть вычисляет оценки релевантности для пар $(q, d+)$ и $(q, d-)$, от которых требуется, чтобы релевантность положительного документа была больше релевантности отрицательного на величину α (обычно $\alpha = 1$).

При обучении модели часто приходится сталкиваться с ситуацией, когда в используемом наборе данных присутствуют только «положительные» примеры (релевантные данному запросу документы), а «отрицательных» примеров нет. Одним из возможных выходов является генерация ненастоящих, случайных пар «запрос — документ», которые и берутся в качестве отрицательных примеров. Этот процесс получил название *майнинга негативных (отрицательных) примеров* (negative mining). Сущность, реализующую процесс майнинга, будем называть *майнером* (miner).

Существует несколько способов майнинга отрицательных примеров:

— Пары «запрос — документ» образуются случайно: для некоторого запроса выбирается случайный документ (или наоборот, документ является настоящим, а запрос берется случайно). Такой подход получил название easy negative mining.

— Для некоторого запроса q выбирается некоторое количество случайных документов $d_1, d_2, ..., d_k$, пары (q, d_i) подаются на вход модели и в качестве отрицательного примера берется пара, для которой предсказание модели максимально (аналогична обратная ситуация, когда для некотрого документа d выбираются запросы $q_1, q_2, ..., q_k$). Этот способ называется hard negative mining [8].

Часто используется промежуточный вариант майнинга: часть примеров образуется посредством easy negative mining, а часть — hard negative mining (так модель учится работать с простыми и сложными примерами).

Для реализации майнинга необходимо решить, откуда брать случайные документы и запросы: источником для майнинга может является текущий батч данных или отдельный источник. Выбор источника отрицательных примеров обычно зависит от имеющихся вычислительных ресурсов, а также от объема используемых для обучения данных.

Как было сказано выше, можно осуществлять майнинг запросов для документа, а можно — документов для запроса. Вопрос о том, какой режим майнинга необходимо выбрать, напрямую зависит от используемой функции потерь: для точечных функций потерь необходимо осуществлять майнинг в обе стороны, а для попарных потерь правильнее майнить документы для запроса (так как в этом случае предсказания модели для документа в паре с разными запросами несравнимы между собой).

1.3 Ранжирование поисковой выдачи

Качество поиска является комплексным понятием и зависит не только от состава документов, отобранных по некоторому запросу, но и от порядка представления информации пользователю. Поэтому ранжирование, позволяющее добиться правильного взаимного расположения документов внутри выдачи, является не менее важной задачей, чем отбор релевантных документов.

Формально *задачу ранжирования поисковой выдачи* можно сформулировать следующим образом. Пусть дана некоторая коллекция документов D и множество поисковых запросов Q. Предположим, что для каждого запроса $q \in Q$ найдено множество документов D_q, соответствующих этому запросу (формируется на этапе поиска). Тогда может быть сформирована выборка $X \subseteq Q \times D, X = \{(q, d) : q \in Q, d \in D_q\}$. Пусть также задано некоторое упорядоченное множество рейтингов Y (чем больше значение, тем выше степень соответствия документа запросу) и заданы оценки релевантности $rel : X \to Y$. Необходимо предложить некоторую ранжирующую функцию $a : Q \times D \to Y$.

Для оценки качества ранжирования обычно пользуются оценками асессоров. Формируется некоторая выборка запросов пользователей и собираются документы из поисковых выдач, порожденных этими запросами. Каждой паре «запрос — документ» асессоры ставят оценку релеватности документа запросу [9]. Исходя из этих оценок вычисляется некоторая метрика качества.

В современных поисковых системах ранжирование поисковой выдачи происходит на основе большого числа факторов различной природы [10]. Поэтому для решения задачи ранжирования часто используется градиентный бустинг: этот метод машинного обучения хорошо работает с данными, структура которых неоднородна. В данной работе для реализации ранжирующей модели была выбрана технология CatBoost, так как данная библиотека обладает рядом преимуществ: высокое качество моделей без необходимости подбора гиперпараметров (согласно результатам сравнения [5] на популярных датасетах, CatBoost выигрывает у открытых аналогов), удобство работы с категориальными признаками, расширяемость, простота использования, быстрота применения.

Основным отличием CatBoost от других реализаций градиентного бустинга является использование *невнимательных решающих деревьев (ODT)*. Согласно работе [11], ODT не только быстро обучаются и применяются, но и устойчивы к изменению параметров с точки зрения итогового качества модели.

1.4 Предложенный метод решения задачи

Как было сказано выше, задача информационного поиска при работе с большими коллекциями документов, как правило, решается в два этапа: отбор кандидатов для формирования выдачи и их ранжирование.

Deep Structured Semantic Model изначально создавалась для решения задачи ранжирования веб-документов [3]: выход модели интерпретируется как оценка релевантности документа запросу и на основании этой оценки и происходит ранжирование. Однако в современных поисковых системах данная модель в основном используется для поиска релевантных документов. При таком подходе ранжирование выдачи осуществляется другой моделью, которая помимо выхода DSSM использует дополнительные факторы (например: время задания запроса, «свежесть» документа, время года, пользовательские предпочтения на основе истории его запросов и др.). Как отмечалось ранее, для решения задачи ранжирования на основе разнородных факторов хорошо зарекомендовал себя градиентный бустинг.

Таким образом, в данной работе рассматривается решение задачи информационного поиска с помощью Deep Structured Semantic Model для поиска релевантных документов и модели CatBoost для ранжирования поисковой выдачи. Особенности моделей позволяют использовать следующий подход:

1. Так как *эмбеддинги* (векторные представления в семантическом пространстве) для документов и запросов строятся двумя разными частями сети, DSSM «разрезается» на запросную и документную части.
2. Используя документную часть DSSM, для документов коллекции строятся их эмбеддинги.
3. При обработке запроса с помощью запросной части модели строится его эмбеддинг, после чего осуществляется поиск ближайших векторов, соответствующих документам коллекции. Отобранные документы и являются кандидатами для формирования поисковой выдачи. При этом для определения близости могут вводиться различные метрики, например: косинусное расстояние, евклидово расстояние или слои нейронной сети, выход которых интерпретируется как расстояние между векторами.
4. Поисковая выдача формируется в результате ранжирования отобранных кандидатов моделью CatBoost.

2 Реализация и обучение моделей

2.1 Предварительная обработка данных

Для работы с моделями используются материалы соревнования TREC Deep Learning Track 2019 [12]. Основным достоинством данного набора данных является его размер: коллекция включает в себя 3213835 документов, а тренировочный датасет содержит 367013 записей «запрос — кликнутый документ». Запросная часть представлена единственным полем, содержащим текст запроса. Документная часть состоит из названия документа, его тела и URL.

Предварительная обработка состояла из следующих этапов:

1. удаление знаков препинания и слов «нецелевых» языков;
2. разбиение текста на предложения;
3. удаление «стоп-слов»;
4. стемминг и лемматизация;
5. разбиение URL на семантические части с помощью специального словаря разделителей.

DSSM получает на вход векторизованные представления запросов и документов (для каждого терма указывается частота встречаемости). В простейшем случае, в качестве термов выступают слова датасета. Однако для больших наборов данных размер такого словаря огромен, что приводит к значительному росту числа параметров модели. Для сокращения размера входного вектора, использовалось хэширование слов: в качестве термов берутся символьные n-граммы, в виде совокупности которых и представляется каждое слово текста.

В данной работе для построения словаря термов использовались следующие подходы: выделение символьных триграмм; использование всех встречающихся в тексте слов (после проведения лемматизации); комбинированный подход — использование ограниченного количества слов и триграмм. Наилучшие результаты показали модели, обученные на векторизациях на основе слов и триграмм, поэтому данный подход был выбран в качестве основного.

Для векторизации URL использовался отдельно обученный векторизатор на основе символьных триграмм. Это обусловлено тем, что состав словаря URL значительно отличается от других текстовых полей.

Отличие подготовки данных для CatBoost заключается в способе векторизации: наилучшее качество показали модели, обученные с использованием идеи хэширования слов со взвешиванием термов на основе TF–IDF.

2.2 Реализация и обучение DSSM для решения задачи поиска

Для реализации модели была выбрана библиотека TensorFlow [13] с использованием Keras API. Это высокоуровневое API включает в себя поддержку специфичной для TensorFlow функциональности, делая работу с данной библиотекой проще без ущерба гибкости и производительности.

Для осуществления майнинга отрицательных примеров реализован класс Miner. Поддерживаются механизмы easy и hard negative mining, а также комбинированный подход. Возможен как майнинг отрицательных документов для запроса, так и отрицательных запросов для документа. Кроме того, Miner умеет работать в поточечном и попарном режимах. Для формирования отрицательных примеров используется отдельный источник данных.

Необходимость майнинга отрицательных примеров делает невозможным использование стандартных средств генерации батчей. С этой целью был создан класс BatchGenerator, поддерживающий встраивание майнера.

В ходе выполнения работы было проведено более 50 экспериментов с архитектурой сети, способом обучения, настройками майнинга отрицательных примеров и метрикой на пространстве эмбеддингов. На основании полученных результатов можно сделать следующие выводы:

— Важную роль играет правильный выбор режима майнинга. Наиболее эффективно в начале обучения генерировать простые примеры, постепенно увеличивая «жесткость» майнера для выявления более сложных закономерностей. Кроме того, при поточечном обучении лучше использовать генерацию отрицательных запросов и документов в обе стороны.
— Нормализация по мини-батчам не только способствует улучшению качества, но и помогает бороться с переобучением сети.
— Не было отмечено прироста качества при использовании Dropout. Кроме того, выяснилось, что использование слоев нормализации после слоев Dropout приводит к ухудшению качества. Наблюдаемые результаты согласуются с работой [14], в которой вопросы сочетаемости данных механизмов рассматриваются с теоретической точки зрения.
— Каскадное соединение слоев не дало ожидаемого прироста качества.
— Модели, обученные с использованием попарного подхода, показывают более высокое качество по сравнению с аналогичными моделями, обученными поточечно.

— Введение собственных метрик в пространстве эмбеддингов не дало желаемого прироста качества: качество моделей, в которых близость вычислялась с помощью дополнительных слоев сети, значительно хуже качества моделей, в которых использовалась косинусная близость.

Наилучшая из обученных моделей сочетает в себе все описанные выше подходы: попарное обучение (функция потерь Triplet-loss), нормализация по мини-батчам, увеличение «жесткости» майнера по мере обучения сети, косинусная близость как оценка релевантности документа запросу. Значение метрики оценки качества отобранных кандидатов Top-1000 accuracy на тестовом множестве составило 98.6%.

2.3 Обучение модели CatBoost для решения задачи ранжирования

Как отмечалось выше, CatBoost позволяет обучать модели, показывающие высокое качество даже при использовании стандартных параметров. Однако настройка параметров модели представляет интерес: эксперименты по подбору гиперпараметров позволили обучить модель с более высоким качеством. Производилась настройка следующих параметров:

— метрика качества обучения;
— количество деревьев;
— скорость обучения;
— глубина деревьев;
— константа $L2$ регуляризации;
— способ построения деревьев.

Для обучения использовались материалы второй части датасета TREC Deep Learning Track 2019 [12]: для каждого запроса представлена выдача из 100 релевантных ему документов.

В данной работе использовалось несколько методов обучения ранжированию, которые отличаются постановкой задачи машинного обучения и выбором функции потерь. Наилучшее качество показала модель, обученная с помощью YetiRank: попарная классификация документов на основе их расположения в выдаче лучше коррелирует с особенностями ранжирования поисковой выдачи. Для оценки качества модели использовалась метрика nDCG. Итоговое качество на тренировочной выборке составило 0.9245, на тестовой выборке: 0.9017.

2.4 Анализ полученных результатов

На основе лучших из обученных моделей реализовано несколько стратегий решения задачи информационного поиска. Для оценки качества поиска использовалась метрика *precision at k*. Результаты замера метрик $precision@1$, $precision@5$, $precision@20$, $precision@100$ представлены в таблице 1.

Таблица 1 – Результаты замера метрик качества поиска

№	Стратегия поиска	$precision@1$	$precision@5$	$precision@20$	$precision@100$
1	Поиск и ранжирование с помощью DSSM	4.0%	22.4%	32.4%	43.3%
2	Ранжирование моделью Catboost без этапа поиска	5.3%	28.6%	42.9%	54.4%
3	Поиск документов с помощью DSSM и ранжирование моделью Catboost	4.7%	26.2%	40.0%	52.1%

На основании проведенных экспериментов можно сделать следующие выводы:

1. Использование DSSM для поиска релевантных документов и их ранжирования является достаточно быстрым, но наименее качественным подходом к решению задачи информационного поиска (модель обучалась для отбора кандидатов, а не для ранжирования поисковой выдачи).
2. Ранжирование документов коллекции моделью CatBoost без предварительного отбора документов показывает наивысшее качество, однако необходимость запуска модели для всех документов делает невозможным применение такого подхода для поиска в большой коллекции.
3. Поиск релевантных документов с помощью Deep Structured Semantic Model и ранжирование выдачи моделью CatBoost по качеству лишь незначительно уступает второй стратегии. В то же время, данный подход эффективен по времени: так как модель CatBoost используется только для ранжирования выдачи, затрачиваемое время в значительной степени определяется скоростью отбора документов.

Таким образом, наилучшие показатели с точки зрения качества поиска и быстродействия показал подход, основанный на двухэтапном решении задачи информационного поиска: поиск релевантных документов с помощью Deep Structured Semantic Model и ранжирование выдачи моделью CatBoost.

ЗАКЛЮЧЕНИЕ

В результате работы решены поставленные задачи.

- Выполнена предварительная подготовка и векторизация данных.
- Проблема отсутствия отрицательных примеров в обучающем наборе данных решена путем реализации рассмотренных механизмов получения отрицательных примеров: easy negative mining и hard negative mining.
- Реализована Deep Structured Semantic Model с использованием фреймворка TensorFlow, а также необходимые для ее обучения и дальнейшего использования инструменты:
    1. майнер отрицательных примеров;
    2. батч-генератор, поддерживающий работу в поточечном и попарном режимах, с возможностью встраивания майнера для генерации отрицательных примеров во время обучения сети;
    3. инструмент разрезания модели на запросную и документную части;
    4. механизм замера метрик качества обученной модели.
- Проведены эксперименты с архитектурой сети, способом обучения, настройками майнинга отрицательных примеров и метрикой на пространстве эмбеддингов.
- Проведены эксперименты по обучению модели CatBoost для решения задачи ранжирования.
- Выполнено сравнение различных стратегий решения задачи информационного поиска. Для лучших из обученных моделей замерены метрики качества поиска, на их основе проанализированы результаты проведенных экспериментов.

Таким образом, в работе предложен способ решения задачи информационного поиска с применением Deep Structured Semantic Model и модели CatBoost. Модели показали высокие значения замеренных метрик качества поиска. Отличительной особенностью описанного подхода является его направленность на работу с большими объемами данных, что позволяет использовать предложенное решение в поисковых системах, требующих не только хорошего качества, но и высоких показателей быстродействия.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 *Manning, C. D.* Introduction to Information Retrieval. / C. D. Manning, P. Raghavan, H. Schutze. — University of Cambridge, United Kingdom: Cambridge University Press, 2008.

2 *Марманис, Х.* Алгоритмы интеллектуального Интернета. Передовые методики сбора, анализа и обработки данных. / Х. Марманис, Д. Бабенко. — СПб.: Символ-Плюс., 2011.

3 *Huang, P.* Learning Deep Structured Semantic Models for Web Search using Clickthrough Data / P. Huang, X. He, J. Gao, L. Deng, A. Acero, L. Heck. — Feb 2013. https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/cikm2013_DSSM_fullversion.pdf.

4 Алгоритм «Палех»: как нейронные сети помогают поиску Яндекса — Блог Яндекса [Электронный ресурс]. — URL: https://yandex.ru/blog/company/algoritm-palekh-kak-neyronnye-seti-pomogayut-poisku-yandeksa (Дата обращения 15.05.2020). Загл. с экр. Яз. рус.

5 CatBoost is a high-performance open source library for gradient boosting on decision trees [Электронный ресурс]. — URL: https://catboost.ai/ (Дата обращения 15.05.2020). Загл. с экр. Яз. англ.

6 *Niu, C.* Structured Semantic Model supported Deep Neural Network for Click-Through Rate Prediction. — 2018.

7 *Schroff, F.* FaceNet: A unified embedding for face recognition and clustering / F. Schroff, D. Kalenichenko, J. Philbin // *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* — Jun 2015. http://dx.doi.org/10.1109/CVPR.2015.7298682.

8 *Li, M.* Deep Instance Level Hard Negative Mining Model for Histopathology Images. — 2019.

9 *Liu, T.-Y.* Learning to rank for information retrieval / T.-Y. Liu // *Foundations and Trends in Information Retrieval.* — 2009. — Vol. 3, no. 3. — Pp. 225–331. http://dx.doi.org/10.1561/1500000016.

10 *Гулин, А.* Яндекс на РОМИП 2009. Оптимизация алгоритмов ранжирования методами машинного обучения. / А. Гулин, П. Карпович, Д. Расковалов, И. Сегалович.

11 *Prokhorenkova, L.* CatBoost: unbiased boosting with categorical features. — 2017.

12 TREC 2019 Deep Learning Track Guidelines [Электронный ресурс]. — URL: https://microsoft.github.io/TREC-2019-Deep-Learning/ (Дата обращения 15.05.2020). Загл. с экр. Яз. англ.

13 TensorFlow. An open source machine learning framework for everyone [Электронный ресурс]. — URL: https://www.tensorflow.org/ (Дата обращения 10.05.2020). Загл. с экр. Яз. англ.

14 *Li, X.* Understanding the Disharmony between Dropout and Batch Normalization by Variance Shift / X. Li, S. Chen, X. Hu, J. Yang // *CoRR*. — 2018. — Vol. abs/1801.05134. http://arxiv.org/abs/1801.05134.