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

РАЗРАБОТКА КЛИКОВЫХ МОДЕЛЕЙ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 451 группы
направления 09.03.04 — Программная инженерия
факультета КНиИТ
Сараева Егора Маратовича

Научный руководитель
доцент, к. ф.-м. н. _____ Б. А. Филиппов

Заведующий кафедрой
к. ф.-м. н. _____ С. В. Миронов

Саратов 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Во многих областях науки есть модели событий. Например, механика имеет дело с твердыми телами и приложенными к ним силами. Модель абстрагирует многие вещи, которые не важны для того, чтобы иметь возможность достаточно точно предсказать исход события. Она позволяет прогнозировать какой-либо процесс без проведения эксперимента, который затем служит лишь как проверка представленной модели. Без использования процесса моделирования исследователям пришлось бы проводить эксперименты каждый раз, когда появляется необходимость предсказать будущие события, что просто невозможно масштабировать.

Когда дело доходит до исследований и разработок в области веб-поиска, проводится множество экспериментов с участием пользователей. Эти эксперименты варьируются от небольших лабораторных исследований до экспериментов с миллионами пользователей реальных поисковых систем в Интернете. Сегодня, как и прежде, принято проводить эксперименты «вслепую», например, пробуя разные представления пользовательского интерфейса веб–поиска и проверяя, какие из них приносят больше кликов и дохода. Однако, исследования в данной области не могли бы продвигаться быстро, если бы у ученых не было некоторого понимания поведения пользователей. Например, пользователи, как правило, предпочитают первый документ последнему документу на странице результатов поисковой системы. Эти знания помогают компаниям создавать алгоритмы ранжирования документов наиболее выгодным для пользователя способом.

Интуитивно понятно, что модель пользователя — это набор правил, которые позволяют моделировать поведение человека с поисковой выдачей в форме случайного процесса. Например, из многочисленных исследований пользователей стало известно, что существует так называемый эффект смещения позиции (position bias [1]): пользователи поисковых систем, как правило, предпочитают документы, находящиеся выше в рейтинге на странице поиска. Существует также смещение внимания к визуально заметным документам (attention bias [2]), смещение внимания к ранее не просмотренным документам (novelty bias [3]) и многие другие типы смещения внимания пользователя, которые пытаются использовать различные модели путем введения случайных величин и зависимостей между ними. Данные отклонения пользовательского поведе-

ния приводят к искажению метрик, необходимых для сравнения поисковых алгоритмов.

Первоначально, такие модели пользователей стали применять при улучшении работы поисковых систем и анализа пользовательского поведения, при выборе результатов запроса. Из–за этого, такой вид моделей называют кликовыми моделями.

В настоящий момент, кликовые модели применяются не только для изучения пользовательского взаимодействия с веб–поиском. Они используются для общего анализа взаимодействия пользователя с поисковой системой — просмотр товаров в интернет магазинах и их покупка, выбор музыки на различных платформах, а также взаимодействие с пользовательским интерфейсом и т. д.

Помимо моделирования пользователей, кликовые модели также используются для улучшения ранжирования документов на странице и для более точного распределения результатов исходя из их релевантности. При этом учитываются и исправляются отклонения пользовательского поведения, которые были описаны ранее (position bias, attention bias, novelty bias и т. д.).

Такой тип моделей имеет огромную область применения, связанную с анализом и изучением пользовательского поведения, а так же с ранжированием результатов по какому–либо запросу.

Многие крупные компании, которые имеют миллионные аудитории, используют различные метрики для анализа трендов в своих поисковых системах. Данные метрики часто бывают подвержены пользовательским отклонениям, что не дает объективно оценивать текущие предпочтения пользователей. Важным аспектом при разработке кликовых моделей является то, что они помогают выявлять и сглаживать подобные отклонения.

Однако, одной из основных проблем при использовании кликовых моделей является то, что в крупных системах они должны хорошо масштабироваться, исходя из объемов поступающих данных или нагрузки. Эта задача становится нетривиальной, когда появляется необходимость в масштабировании всего процесса обучения, дообучения и дальнейшей эксплуатации результатов модели. Каждый этап, который проходят данные, должен быть оптимизирован и готов к горизонтальному или вертикальному масштабировании исходя из поставленных требований.

Цель настоящей работы — создание масштабируемого и распределенного приложения, используемого для обучения кликовых моделей. Архитектура приложения рассчитана на быструю обработку большого объема данных при обучении кликовой модели и ранжировани результатов веб–запроса на странице поиска.

В связи с поставленной целью **задачами** преддипломной практики являются:

- Разработать метод распределенного обучения кликовой модели;
- Разработать методы дообучения кликовых моделей с использованием новых данных;
- Реализовать приложение, включающее в себя масштабируемую версию одной из кликовых моделей и инфраструктуру, обеспечивающую полноценную работу данной модели;
- Сравнить распределенную и однопоточную реализацию кликовых моделей.

Структура и объём работы. Бакалаврская работа состоит из введения, 3 разделов, заключения, списка использованных источников и 4 приложений. Общий объем работы — 72 страниц, из них 61 страница — основное содержание, включая 15 рисунков, список использованных источников информации — 20 наименований.

1 Теоретические основы задачи

1.1 Терминология

Кликовые модели описывают поведение пользователя на странице результатов поисковой системы. Кликовые модели рассматривают этот процесс при поиске как последовательность наблюдаемых и скрытых событий. Каждое событие описывается двоичной случайной величиной X, где $X = 1$ соответствует наступлению события, а $X = 0$ означает, что событие не произошло.

Основными событиями, которые учитывают большинство кликовых моделей, являются следующие:

— E: пользователь проверяет объект в поисковой выдаче;
— A: пользователя привлекает представление объекта поиска;
— C: пользователь совершает клик по объекту;
— S: информационная потребность пользователя удовлетворена.

Кликовые модели определяют зависимости между этими событиями и нацелены на оценку полных или условных вероятностей соответствующих случайных величин, например, $P(E = 1)$, $P(C = 1|E = 1)$ и т.д.

1.2 Позиционная кликовая модель (PBM)

Позиционная кликовая модель основана на гипотезе проверки(examination): $C_u = 1 \Leftrightarrow E_u = 1$ и $A_u = 1$

Это означает, что пользователь нажмет на документ тогда и только тогда, когда он прочитает небольшой отрывок текста(этот процесс называется examination) и он его привлечет(attractive). [4]

Привлекательность здесь является характеристикой фрагмента документа, а не его полного текста. Обычно, пользователь может прочитать этот фрагмент на поисковой странице, он располагается ниже заголовка очередного результата. Было выявлено, что этот параметр коррелирует с полнотекстовой релевантностью.

Вероятность того, что пользователь изучит документ, сильно зависит от его ранга или позиции в поисковой выдаче и, обычно, уменьшается с повышением ранга.

1.3 Алгоритмы обучения кликовых моделей

Чаще всего, при обучении кликовых моделей используется процесс изучения параметров кликовой модели на основе прошлых наблюдений за клика-

ми. Этот процесс называется оценкой параметров или изучением параметров.

Существует два основных метода, используемых для оценки параметров моделей кликов, а именно:

— алгоритм оценки максимального правдоподобия (MLE);
— алгоритм максимизации математического ожидания (EM)

1.4 EM алгоритм

Алгоритм максимизации математического ожидания (EM) может быть использован для оценки значения параметра Θ распределения Бернулли, если параметр X или любой из его «предков» является скрытым параметром.

Алгоритм EM работает путем итеративного выполнения операции математического ожидания (E) и операции максимизации (M). На этапе E предполагается, что Θ известно и фиксировано.

Существует теоретическая гарантия сходимости, которая означает, что значение правдоподобия будет улучшаться на каждой итерации алгоритма и в конечном итоге сойдется к (обычно локальному) максимуму.

Однако, быстрая сходимость данного алгоритма не гарантируется и при использовании EM на практике, обычно, либо выполняется фиксированное количество итераций, либо отслеживается разница значений алгоритма между двумя итерациями и, если она стала меньше, чем фиксированное значение ϵ, то выполнение алгоритма прекращается. [5]

1.5 Распределенные системы

В современном мире процесс обучения и применения ML моделей тесно связан с обработкой больших объемов информации. Современные компании обучают свои модели поиска и ранжирования на петабайтах кликовых данных.

В таком случае, первоочередными становятся задачи передачи, хранения и взаимодействия с данными. Важное место в этом процессе занимает отказоустойчивость и быстродействие всей системы.

Распределенная система — это совокупность сервисов, которые работают вместе, образуя единую экосистему данных для конечного пользователя.

2 Практическая часть работы

В практической части данной работы будут представлены две реализации позиционной кликовой модели PBM. Обучение происходит с помощью EM алгоритма. Первый вариант модели является однопоточным Python приложением. Второй вариант представляет из себя распределенную архитектуру, состоящую из web–сервиса, генератора данных, сервиса доставки и обработки сообщений Kafka и распределенного хранилища данных Hadoop. Модель реализована с помощью фреймворка больших данных Spark на языке программирования Scala.

2.1 Распределенная реализация кликовой модели

Ранее описанная реализация кликовой модели имеет несколько недостатков. Во–первых, все вычисления выполняются последовательно в одном потоке, что сильно снижает производительность самой модели и повышает время ее обучения. Во–вторых, однопоточная реализация не может быть обучена на объеме данных, который превышает оперативную память машины, на которой выполняется программа.

Распределенная реализация использует набор различных инструментов обработки и передачи данных. PBM модель реализована с использованием фреймворка Apache Spark. Необходимые для корректной работы приложения данные хранятся в распределенной файловой системе Apache Hadoop. Сервис, который отвечает за генерацию ответов на пользовательские запросы, использует технологию Flask. Пользовательские поисковые сессии вместе с результатами доставляются в файловую систему посредством технологий Apache Kafka и Kafka Connect.

2.2 Реализация класса модели

Взаимодействие с моделью начинается с ее обучения, на заранее очищенных и подготовленных данных. В качестве источника наблюдаемых пользовательских сессий данная реализация может использовать parquet файлы, json файлы, DataSet и DataFrame объекты.

Были добавлены функции сериализации и десериализации модели необходимы для ее дообучения. Модель сериализуется в хранилище данных HDFS в виде parquet файлов, представляющих из себя DataFrame объект с обученными параметрами модели.

2.3 Описание распределенного алгоритма обучения

Распределенный алгоритм обучения PBM модели отличается от последовательного тем, что данные из тренировочного набора равномерно распределяются по исполнителям кластера, а затем собираются обратно. Такого эффекта можно достичь не только благодаря фреймворкам распределенной обработки данных, но и с помощью обычного многопоточного подхода.

Важной особеностью такого процесса является то, что данные можно распределить по процессам исходя из значений Query–Result–Rank. Эти параметры являются статичными, что уменьшает количество обменов данными между процессами при итеративном обучении.

2.4 Реализация алгоритма обучения

Обучение описанной выше PBM модели происходит с помощью EM алгоритма, реализованного в контексте Spark приложения. Основная идея данного подхода состоит в том, что все наблюдаемые пользовательские сессии хранятся в виде DataFrame объекта.

Параметры модели инициализируются значениями 0.5, которые хранятся в виде числителя и знаменателя для избежания накопления ошибки при операциях с плавающей точкой на этапе итеративного обучения.

2.5 Способы дообучения модели

дной из важных особенностей данной реализации PBM модели является возможность ее дообучения на новых данных. Было разработано два способа дообучения модели — один основан на умножении уже обученных данных на константу меньше 1 и обучения новой модели с использованием объединения новых и полученных данных, другой основан на полном дообучении модели на очищенных данных.

2.6 back–end сервер.

В цикле существования модели важное место занимает способ доставки наблюдаемых пользовательских сессий. Эту ответственность возлагает на себя back–end сервер, реализованный с помощью технологии Flask.

В данном сервере существует два обработчика запросов. Один из них отвечает за обработку GET запросов и возвращает идентификаторы документов, являющихся результатами пользовательского запроса. Другой отвечает за

прием записей о пользовательских сессиях вместе с информацией о кликах и помещение полученных данных в Kafka topic. [6]

2.7 Генератор данных.

Генерация данных является внешним для всей системы процессом. Данное приложение моделирует поведение пользователя, подчиняющееся определенным правилам. Генератор осуществляет запросы к серверу, получает документы, как результаты запроса, моделирует клики и отправляет ответ серверу.

При генерации кликовых данных следует учитывать, что вероятность клика по странице складывается из вероятности проверки и вероятности привлекательности. Вероятность проверки является статичной и зависит от позиции документа. Это значит, что результирующая вероятность для каждой позиции будет рассчитана как произведение вероятностей проверки и привлекательности.

2.8 Настройка брокера сообщений Kafka

Ранее описанные сервисы обеспечивают генерацию и обработку пользовательских данных. Но для того, чтобы все сервисы работали в единой системе необходимо настроить инфраструктуру, отвечающую за хранение и передачу данных. Вся инфраструктура построена с использованием технологии виртуализации Docker.

Kafka сервис необходим для доставки сообщений от back–end сервера до распределенного хранилища. Данный сервис обеспечивает отказоустойчивость и гарантирует сохранность данных. Наблюдаемые пользовательские клики загружаются в раздел, хранящийся на Kafka сервере. [7]

Помимо Kafka сервера в данном приложении применяется еще одна технология, обеспечивающая real–time обработку сообщений в Kafka. Данная технология называется KafkaStreams и является инструментом стриминговой обработки информации.

Последним шагом в процессе передачи сообщений с использованием Kafka является их сохранение в распределенное файловое хранилище Apache Hadoop. За это отвечает технология Kafka Connect. Она работает как отдельный сервер или набор серверов. Они подключаются к серверам Kafka и читают данные из топиков. Kafka Connect имеет множество встроенных видов коннекторов и может обеспечивать как считывание данных так и запись сообщений.

3 Тестирование моделей

Для тестирования созданных моделей проводились испытания по их обучению с разным количеством итераций и разным объемом данных. Данные вероятности были взяты из исследований компании Google относительно распределения вероятности кликов по позициям на SERP. [8] Было сгенерировано 20000 пользовательских сессий.

Основными целями сравнения были производительность моделей: затраченное время для разного количества итераций и для разного количества данных, а также, значения, полученные после обучения моделей.

Однопоточная модель и распределенная модель PBM работают практически идентично. Разница в значениях не превышает 0.03 по каждому из рангов. Вероятности кликов практически совпадают с теоретической вероятностью, заданной при генерации данных.

Было исследовано два метода дообучения. Один из них основан на фильтрации всех имеющихся данных и выбор самых новых значений. Способ дообучения, использующий все имеющиеся данные, будет работать точнее, однако, способ дообучения кликовой модели с использованием compaction работает быстрее, так как выделяется меньший набор данных.

Следующее исследование связано с временем выполнения, затраченным на обучение каждой из моделей. Тестирование проводилось на нескольких наборах данных — от 50 MB до 500 MB.

Однопоточная модель, реализованная на языке Python, работает лучше, для небольшого количества данных. Однако, реализация PBM модели с использованием фреймворка Spark начинает работать лучше для большего количества данных. Это оправдано тем, что Spark, помимо необходимых вычислений, производит операции по координации работы на worker узлах, а также, поддерживает метаданные для всех операций.

ЗАКЛЮЧЕНИЕ

В заключении проведенного нами исследования можно сделать следующие основные выводы по теме.

Кликовые модели — процессы, моделирующие поведение пользователя в какой–то поисковой системе. Также, они применяются для ранжирования результатов какого–либо запроса и исправления метрик при проявлении аномалий вида position bias, attention bias и т. д.

В настоящее время область применения кликовых моделей является практически безграничной. Их можно использовать при создании интернет магазинов, аудио сервисов, онлайн кинотеатров или поисковых систем. Все эти области объединяет необходимость изучать пользовательское поведение на странице и прогнозировать его будущие шаги для создания положительного пользовательского опыта.

На момент написания данной работы не существует доступных открытых реализаций распределенных кликовых моделей. Все крупные компании используют собственные реализации моделей и алгоритмов их обучения, оптимизированные под конкретные поисковые системы.

Результатом данной работы стало создание двух видов позиционной кликовой модели — одна из них выполняет вычисления в одном потоке, а другая является распределенной. Также, распределенная PBM модель была интегрирована в систему, необходимую для ее непрерывного обучения и использования. Было разработано и исследовано два вида дообучения модели — один из них основан на фильтрации и выборке новых данных, а другой — на использовании констаны для снижения веса старых данных.

За выполнение функции генерации страниц и передачи ответов от клиентов отвечает back–end сервер. Запросы к серверу осуществляет генератор данных, который позволяет гибко задавать правила, моделирующие поведение пользователя в поисковой системе.

В роли хранилища данных, необходимых для обучения, выступает распределенная файловая система HDFS. За доставку и обработку пользовательских ответов отвечают такие технологии как Apache Kafka, Kafka Streams, Kafka Conenct. Распределенная модель реализована с помощью фреймворка Apache Spark. Нагрузка на сервер моделировалась с помощью генератора синтетических данных.

При сравнительном анализе двух реализаций позиционной кликовой модели было выявлено, что однопоточная реализация работает быстрее на небольших объемах данных, в то время как Spark реализация лучше оперирует с большими объемами информации.

Также, исходя из проведенных наблюдений, можно сделать вывод, что обе модели дают идентичные предсказания вероятностей кликов с точностью до 0.03. Эти результаты совпадают с теоретической вероятностью, заданной при генерации данных, и, следовательно, можно утверждать, что обе модели работают корректно.

При исследовании методов дообучения модели был сделан вывод, что метод экспоненциального угасания работает точнее, так как использует весь имеющийся объем данных. Тем временем, метод, использующий выборку новых значений, работает быстрее.

На основании проведенных исследований были выявлены преимущества использования распределенной реализации позиционной кликовой модели по сравнению с ее однопоточной версией.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 *Guan, Z.* An eye tracking study of the effect of target rank on web search. / Z. Guan, E. Cutrell // *CHI '07: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. — 2007. — Vol. 875, no. 4. — Pp. 417–420.

2 *Chen, D.* Beyond ten blue links: enabling user click modeling in federated web search. / D. Chen, W. Chen, H. Wang // *WSDM '12*. — 2012. — Vol. 553, no. 10. — Pp. 463–472.

3 *Zhang, Y.* User-click modeling for understanding and predicting search-behavior. / Y. Zhang, W. Chen, D. Wang // *KDD '11*. — 2011. — Vol. 1414, no. 9. — Pp. 1388–1396.

4 *Craswell, N.* An experimental comparison of click position-bias models. / N. Craswell, O. Zoeter, M. Taylor, B. Ramsey // *WSDM '08*. — 2008. — Vol. 1465, no. 8. — Pp. 87–94.

5 *Dempster, A. P.* Maximum likelihood from incomplete data via the EM algorithm. Journal of the royal statistical society. / A. P. Dempster, N. M. Laird, D. B. Rubin. — Санкт-Петербург: Wiley, 1977. — 38 pp.

6 Flask API Documentation [Электронный ресурс]. — URL: https://flask.palletsprojects.com/en/2.1.x/ (Дата обращения 21.04.2022). Загл. с экр. Яз. анг.

7 *Наркеде, Н.* Kafka: The Definitive Guide / Н. Наркеде. — Sebastopol, CA: O'Reilly Media, 2017. — 322 pp.

8 Click probabilities in the Google SERPs [Электронный ресурс]. — URL: https://www.sistrix.com/blog/click-probabilities-in-the-google-serps/ (Дата обращения 15.03.2022). Загл. с экр. Яз. анг.