МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

КУРСОВАЯ РАБОТА

по дисциплине «Производственная практика НИР»

Тема: Машинное обучение на графах

Студент гр. 8304 _____ Кирсанов А.Я.

Преподаватель _____ Заславский М.М.

Санкт-Петербург

2023

ЗАДАНИЕ
НА КУРСОВУЮ РАБОТУ

Студент Кирсанов А.Я.

Группа 8304

Тема работы: Машинное обучение на графах

Исходные данные:

Формулирование постановки задачи, описание системы для работы с Janus Graph

Содержание пояснительной записки:

«Содержание», «Введение», «Постановка задачи», «Результаты работы в осеннем семестре», «Описание предполагаемого решения», «План работы на весенний семестр», «Список использованных источников»

Предполагаемый объём пояснительной записки:

Не менее 10 страниц.

Дата выдачи задания: 24.11.2023

Дата сдачи реферата: 20.12.2023

Дата защиты реферата: 22.12.2023

Студент _____ Кирсанов А.Я.

Руководитель _____ Заславский М.М.

АННОТАЦИЯ

В данной работе рассмотрен подход к реализации машинного обучения на графах с помощью библиотеки машинного обучения Deep Graph Library. Разрабатываемая система даёт возможность использовать машинное обучение для создания прогнозов с использованием данных на графе. Система автоматизирует работу по выбору и обучению наилучшей модели ML на графе напрямую, используя внутреннюю API. В результате можно создавать, обучать и применять ML на данных Janus Graph в разы быстрее без необходимости вручную выгружать данных и применять их для обучения на своих нейронных сетях.

SUMMARY

In this paper, we consider an approach to implementing machine learning on graphs using the Deep Graph Library machine learning library. The system under development makes it possible to use machine learning to create forecasts using graph data. The system automates the work of selecting and training the best ML model on the graph directly using the internal API. As a result, you can create, train and apply ML on Janus Graph data many times faster without having to manually upload data and apply it to training on your neural networks.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Одним из главных преимуществ графовых баз данных является их универсальность, то есть возможность хранить в них и реляционные и документальные и сложные семантические данные. При этом сама модель построения БД может меняться и модифицироваться в процессе развития приложения без изменения архитектуры и исходных запросов.

При накоплении большого объёма данных на графе можно получить возможность их анализировать и делать на их основе предсказания. Например, имея граф, содержащий информацию об аэропортах мира, расписании полётов и начальных и конечных точек маршрута самолётов, можно делать предположения о том в какой момент откуда и куда полетит самолёт, о его типе и продолжительности полёта. Другим примером может быть граф списка покупок пользователей на той или иной онлайн площадке. По нему можно сделать предположении о предпочтении пользователя, основываясь на его возрастной группе и сделанных им ранее покупках.

В этом случае можно реализовать систему, применяющую машинное обучение на графе для выполнения тех или иных предсказаний. Так можно предсказывать свойства узла графа, возможное наличие связей между узлами.

ПОСТАНОВКА ЗАДАЧИ

Актуальность:

На данный момент существует только один сервис, способный создавать и обучать модели машинного обучения, основываясь на данных из Janus Graph, это – Neptune ML (Amazon). Так как эта система является во-первых платной, во-вторых американской, использовать её в отечественной разработке официально не представляется возможным. Потребность в такой системее на данный момент имеется у компании БалтИфноКом для использования в своём программном продукте – графовой СУБД Октопус.

Объектом исследования являются подходы к созданию систем машинного обучения на графах.

Предметом исследования являются графовые БД.

Целью работы является создание аналога сервиса Neptune ML, позволяющего производить машинное обучение на графовой БД Janus Graph.

Задачами работы являются:

- Рассмотреть принцип работы Neptune ML
- Найти open-source аналоги компонентов Neptune ML (код Neptune ML является проприетарным)
- Реализовать функционал сервиса для работы с Janus Graph.

1. РЕЗУЛЬТАТЫ РАБОТЫ В ОСЕННЕМ СЕМЕСТРЕ

1.1. План работы на осенний семестр

На начало семестра был поставлен план – понять принцип работы Neptune ML, найти open-source аналоги компонентов и создать рабочий прототип сервиса для интеграции с Janus Graph.

1.2. Принцип работы Neptune ML

На рисунке 1 представлена схема, по которой работает Neptune ML.

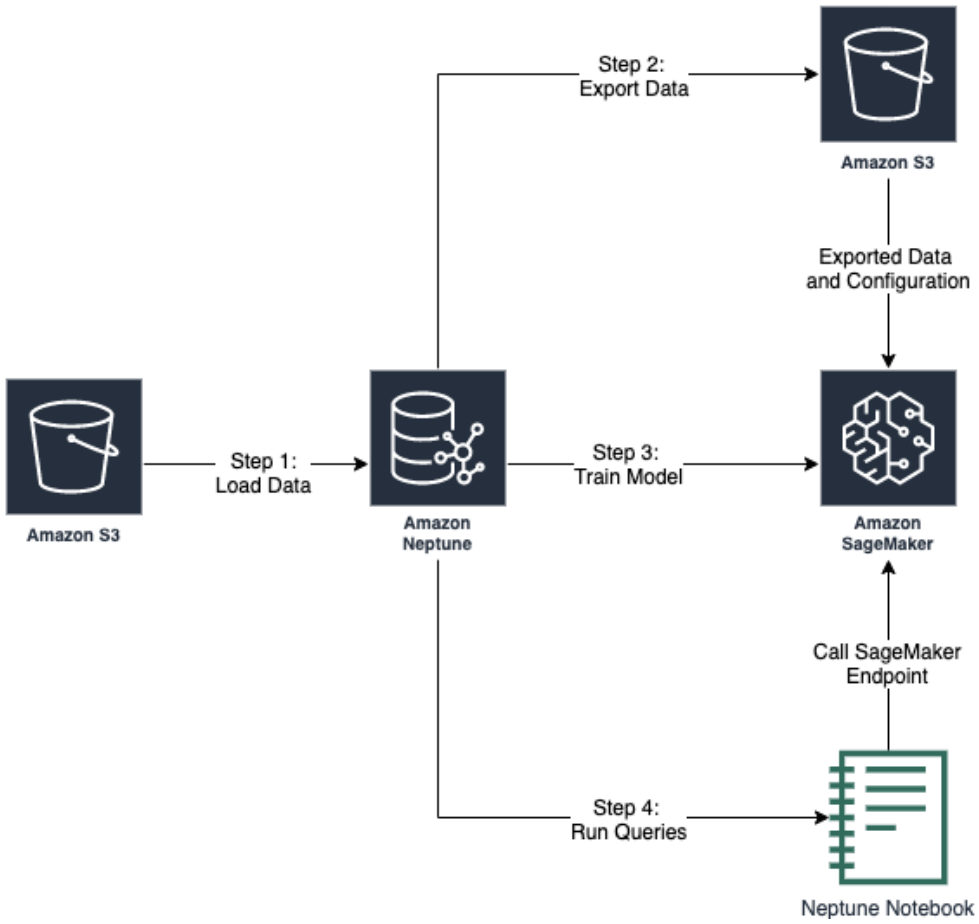

Рисунок 1 – Принцип работы Neptune ML

Данные выгружаются из Amazon S3 бакета в Amazon Neptune. Далее данные экспортируются в другой бакет Amazon S3, из которого в последствии перейдут в Amazon SageMaker.

SageMaker это сервис Amazon для машинного обучения. Он использует библиотеку Deep Graph Library для обеспечения высокопроизводительного машинного обучения для любого варианта использования. С помощью

SageMaker можно создавать, обучать и развертывать модели ML. В нём можно использовать такие инструменты как Jupyter Notebook, отладчики, профайлеры, MLOps.

В SageMaker на шаге 2 подаются данные графа, на шаге 3 из Neptune подаётся нужная модель для обучения, на шаге 4 через Amazon Neptune пользователь запускает нужный запрос через Neptune Notebook, который вызывает Endpoint SageMaker и запускает процесс обучения. Результаты приходят в Neptune Notebook.

1.3. Open-source аналоги компонентов

Neptune ML использует для машинного обучения open-source библиотеку Deep Graph Library. Эта библиотека позволяет производить машинное обучение на графах, имеет соответствующие модели для представления графа, работы с ним и подаче графа в модель для обучения.

В простом случае граф – его узлы, связи и свойства связей векторизуются с использованием text2vec.

Open-source аналогом самого графа служит JanusGraph – он позволяет хранить множество узлов и связей разных типов и иметь к ним доступ извне, например из СУБД Октопус или Jupyter Notebook.

Таким образом, основные open-source сервисы для реализации ML Flow это JanusGraph и DGL. В качестве SageMaker решено было использовать Python скрипт, запускать который можно будет из Октопуса. В качестве Neptune ML можно использовать сам Октопус.

1.4. Создание рабочего прототипа сервиса для интеграции с Janus Graph.

Версия прототипа, созданная внутри компании представляет собой набор сервисов и связей между ними, в совокупности реализующих поведение Neptune ML.

Endpoint service представляет собой Docker образ, доступный на порту 8081. Этот сервис принимает запросы для обучения датасета.

Export service представляет собой Docker образ. Он позволяет распарсить задачу Октопуса по экспорту данных и экспортировать узлы графа в csv файл для последующей загрузки графа в DGL. На этих данных обучается модель и производится предсказание нужных параметров модели.

JanusGraph – сервис, разворачивающийся в Docker контейнере, содержащий граф, с которым работает пользователь. Это тестовый сервис, который впоследствии будет заменён JanusGraph, разворачиваемым Октопусом.

Processing service – Docker образ, слушающий endpoint s3 и выполняющий векторизацию данных из графа.

Testing service – сервис (Docker образ), использующий тестовые Jupyter ноутбуки, из которых могут быть посланы запросы к другим сервисам для экспорта и обучения на конкретных данных.

Train service – сервис, использующий библиотеку DGL для обучения модели и предсказания на ней.

Со стороны пользователя прототип работает как Jupyter Notebook. В нём можно заполнить граф нужными данными и с помощью gremlin запроса запросить предсказание определённых свойств узла, основываясь на текущих данных.

2. ОПИСАНИЕ ПРЕДПОЛАГАЕМОГО РЕШЕНИЯ

На текущем этапе необходимо интегрировать сервис с Октопусом. Для этого требуется реализовать соответствующие задачи в октопусе, позволяющие вызывать нужные endpoint-ы из интерфейса Октопуса. На данный момент предполагается, что это будет выглядеть как клик по узлу и выбор нужной задачи из списка, например «Определить значения недостающих свойств» в том случае, если не все свойства узла заполнены. В этом случае будет послан запрос с параметрами задачи на endpoint service, который сможет выполнить обучение и предсказание на модели и вернуть ответ Октопусу, который сможет заполнить на своей стороне узлы графа. То есть предполагается всю работу, связанную с экспортом из Janus Graph и дальнейшей работой с ним передать Октопусу, для этого возможно переписывание некоторой части кода с Python на Java.

В дальнейшем возможно разворачивание сервисов на локальных машинах компании для удобства работы с ними от любого клиента Октопуса.

Также предполагается использование предобученных моделей для особо больших проектов некоторых заказчиков.

ПЛАН РАБОТЫ НА ВЕСЕННИЙ СЕМЕСТР

Планируется доработать сервис, который на данный момент работает в тестовом режиме. Вместо Jupyter ноутбуков предполагается использовать задачи из Октопуса.

Планируется переписать сервис в более корпоративном ключе и, возможно реализовать экспорт данных не через csv файл, а на ходу, что может потребовать прямой интеграции кода, написанного на Python и на Java. Либо использовать батчинг, так как экспорт большого объёма данных может занимать большое количество оперативной памяти. В таком случае придётся переписать сервис для обучения модели батчами.

Помимо технических работ предполагается написание пояснительной записки по ходу разработки.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Lingfei Wu. Graph Neural Networks: Foundations, Frontiers, and Applications.: Springer Singapore, 2022. 725 с.
2. Franco Scarselli. The Graph Neural Network Model // IEEE Transactions on Neural Networks. 2009, вып. 20. С. 61-80.
3. Alessio Micheli. Neural Network for Graphs: A Contextual Constructive Approach // IEEE Transactions on Neural Networks. 2009, вып. 20. С. 498-511.
4. Sanchez-Lengeling. A Gentle Introduction to Graph Neural Networks // Distill. 2021, вып. 9. С. 15-31.
5. Deep Graph Library // DGL. URL: https://www.dgl.ai/ (дата обращения: 18.12.2023).
6. AWS Documentation // AWS. URL: https://docs.aws.amazon.com/ (дата обращения: 18.12.2023)
7. Amazon Neptune // AWS. URL: https://aws.amazon.com/neptune/ (дата обращения: 18.12.2023)
8. Amazon SageMaker // AWS. URL: https://aws.amazon.com/sagemaker/ (дата обращения: 19.12.2023)
9. Stanford Large Network Dataset Collection // Stanford University. URL: https://snap.stanford.edu/data/ (дата обращения: 19.12.2023)
10. Justin Gilmer. Neural Message Passing for Quantum Chemistry // Proceedings for Machine Learning Research. 2017, вып. 8. С. 1263-1272.