Oğuzhan Güngör

24411006

INF 528 ADVANCED TOPICS IN COMPUTER ENGINEERING FINAL PROJECT REPORT

# PART I – Administrative Features and Management Aspects

## Project Title

Building a Knowledge Graph enriched with Large Language Models and Linked to DBpedia.

## Team

Oğuzhan Güngör – Presenter

Oğuzhan Güngör – Data preparation & data preprocessing

Oğuzhan Güngör – Knowledge graph design and enrichment

## Github

<https://github.com/theFellandes/LLM-Final-Project>

# PART II – The Study

## Abstract

The goal of this project is to enrich knowledge graph using DBpedia and LLMs. Key finding is that 2 million data is a huge data and handling it requires alternative approaches. Utilizing DBpedia and LLMs are good way to ensure data quality.

## Introduction

This section aims to address the problem or challenge at hand, state the project's objectives. In this project the goal was create a knowledge graph using the **Goodreads Book Datasets with User Rating 2M[i]**. With this dataset and **DBpedia[ii]** and LLMs the goal is to create a knowledge graph that generates book suggestions in a clever way and observe which LLM performs better in comparison to others.

## Materials and Methods

This section describes the tools, technologies, and methodologies used in the project's design and implementation. For the baseline of the project, Python and Neo4j has been used. The backend, data manipulation, querying all made possible using Python. Utilizing a Docker container containing Neo4j has been the solution for the database storage and initialization. Downloading the dataset made possible with Python. Using Kagglehub, the dataset was downloaded and stored into objects to be sent to Neo4j. After that, the objects have been stored inside the Neo4j database. For storing nodes, it was also possible to utilize APOC plugin however due to race condition, APOC plugin returned errors therefore utilizing a multiprocess approach was more suitable for the data integrity. Then APOC plugin’s script has been changed for better data handling and easier transferring however this also had data integrity issues but 0 issue approach was time costly therefore abandoned. Also, for improving the knowledge graph, utilizing DBpedia’s database was tinkered with. Utilizing DBpedia for obtaining Genre and Awards improved database’s quality. For sentiment analysis, LLMs were used and compared with each other. Mainly BART and OpenAI were used for sentiment analysis.

## Results

This section presents the findings. Utilizing multiprocessing instead of concurrency prove easier to handle due to connection pooling issues that arises with dockerized Neo4j. Therefore, storing whole database took too much time, around 16 hours. This time sink was also due to host computer’s SSD having less storage capacity.

In the sentiment analysis part, BART and OpenAI models were used and their results were similar to each other. When it comes to positive or negative, both performed really good. Alas, OpenAI model was better when it comes to scoring 1 to 5.

## Conclusion

Utilizing knowledge graph with Sparql and LLMs improves knowledge graph’s knowledge significantly. Performing sentiment analysis using BART and OpenAI shows that in case of good or bad, both performs similarly, however in the case of 1 to 5, OpenAI performs significantly better.

## References

1. <https://www.kaggle.com/datasets/bahramjannesarr/goodreads-book-datasets-10m/data>
2. https://www.dbpedia.org/