Master Thesis

Leveraging university curricula and course descriptions to augment a knowledge graph with degree-skill relationships

Saloni Singh

a thesis submitted in partial fulfilment of the requirements for the degree of

MA Linguistics
(Text Mining)

Vrije Universiteit Amsterdam

Computational Lexicology and Terminology Lab
Department of Language and Communication
Faculty of Humanities

Supervised by: Jose Angel Daza
2^{nd} reader: Lucia Donatelli

Submitted: August 15, 2023

Abstract

This thesis project aimed to explore how university curricula and course descriptions can be leveraged to improve the quality of degree-skill relationships in knowledge graphs. Text data were collected from universities across Europe and analyzed using frequency statistics, the TDIDF, and the Textrank algorithm. This analysis was used to mine connections between degree names and skill lists. The results of the research demonstrated improvements in the precision of skill ranking within degrees, thus using it to augment the graph. Then using natural language prompts and triple-based prompts for graph pruning, the accuracy of the top 7 skills was improved in the final approach with the test set coming up to 84%.

Declaration of Authorship

I, Saloni Maninder Singh, declare that this thesis, titled *Leveraging university curricula and course descriptions to augment a knowledge graph with degree-skill relationships* and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a degree degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Date: 11-08-2023

Signed: Saloni Singh

Acknowledgments

I would like to thank my supervisors Jose Angel Daza and Lucia Donatelli for the constructive suggestions, guidance, and understanding that they provided me with during the entire duration of this thesis project. I would like to extend my gratitude to the team of Textkernel for creating a stimulating and positive working environment and for believing in the project and in me, by providing all the necessary resources for the success of this work. I am also grateful to all the CLTL staff of the VU Amsterdam for the invaluable knowledge that they passed on during the course. Finally, a special thanks to my family for the invaluable support, their love, and the life lessons that brought me here. To them I owe the realization of any of my ambitions.

List of Figures

Contents

Chapter 1

Introduction

Human resource management (HRM) is a vital aspect of organizational success since they manage its most valuable asset: people. They ensure that the employees are performing at their best and contributing to the organization's objectives. HRM, therefore, plays a crucial role in attracting and retaining talent by creating a positive work environment, offering competitive compensation and benefits, and providing opportunities for career growth and development. However, many HR tasks are routine and repetitive, requiring significant time and resources to complete. Automating these tasks can offer several benefits to organizations, including increased efficiency and cost savings. By using automated systems, HR departments can provide candidates with a more efficient and personalized recruitment experience, improving their overall impression of the organization and increasing the likelihood of attracting top talent.

Natural Language Processing (NLP), is important in this context because job description data and resume data mostly come as text. It can help HR departments to analyze and understand the language used in job descriptions, resumes, and other text-based documents. Information extraction techniques can automatically extract relevant information from job descriptions and resumes, such as job titles, skills, and experience, and match them with candidate profiles. In this thesis, we will explore the potential of NLP techniques to cater to recent graduates, who represent a valuable talent pool for organizations.

Entry-level professionals bring several advantages to organizations, including building a talent pipeline, promoting diversity and inclusion, and driving innovation and creativity. However, fresh graduates may not write enough skills on their CVs for several reasons, including a lack of work experience, uncertainty about what to include, and a lack of understanding of their own skills. Therefore, identifying the skills of fresh graduates can be challenging but crucial, to match them with the appropriate job applications.

1.1 Problem definition

This thesis project is in collaboration with Textkernel (TK). "Textkernel's parsing, search and matching have gone through rigorous testing and implementation by global staffing and recruitment agencies, worldwide corporate HR organizations and top-tier management consulting organizations." Textkernel (2023) TK has a very accurate, hand-curated knowledge graph (KG) that supports their resume parser. It has a large taxonomy of professions and skills and their interconnections. e.g: a profession-type

node with the identity *"Front End Developer"* would have multiple skill-type nodes associated with it, like - *"Angular"*, *"node.js"*, *"Javascript"*, etc. These associated skills are at the concept level of skill names. A concept-level skill name will have multiple surface forms associated with it across languages and synonyms. This data is stored as a child-parent tree hierarchy graph. e.g.- Natural Language Processing is the parent and could have *"NLP"*, *"NLP techniques"*, *"text mining"* etc as its children. Similarly, there is a tree graph for all the professions the company has identified, and their surface forms in various languages/ synonyms. The list of core concepts for both these node types was curated by the company's linguist and business team along with their clients as per company requirements. There are 5000 professions and 13000 skills in the company's graph, with 15 synonyms on average.

When TK receives a CV from a client, their parser picks up many details about the candidate, the most important of which is their skills. They normalize the skills they extracted from the CV using the taxonomy described above. They also pick up job titles from the CVs. They use the profession-skill relations in the taxonomy to postulate other skills the candidate may have- based on the job title and what skills that profession needs on average- but have not written about explicitly in the CV. This model of deducing unwritten skills of professionals based on the high-quality knowledge graph is what has put Textkernel in the lead in their sector. TK's research showed that a similar approach for freshly graduated students and their educational degree names would boost the performance of their systems in key market areas. I will therefore add a third type of node entity to the graph: degrees (master in psychology, law, bachelor in city planning, etc.), with appropriate synonyms/ surface forms for each core degree, and relevant connection to the skills learned by the students of that program on average.

This research project holds several potential benefits. Improved candidate matching will give TK a competitive advantage since the system could be used to suggest courses to candidates who lack specific skills required for their desired jobs or who wish to up-skill in general. This ability to support and leverage the skills of fresh graduates demonstrates a commitment to their professional growth and improves an organization's reputation as an employer of choice. Organizations can strengthen their workforce by attracting and retaining top talent and driving innovation and success. This thesis project will contribute to the broader field of HRM and NLP. The successful implementation of the proposed system can pave the way for future research and applications in talent management and recruitment processes.

1.2 Research question and solution

1.2.1 Research Question

Based on the problem statement given above, the research question that arises for this thesis is -

"How can university curricula and course descriptions be leveraged to improve the quality of degree-skill relationships in knowledge graphs, and what methods can facilitate the extraction and enrichment of this information?"

Sub questions -

- How can the enriched graph be evaluated for its effectiveness in matching skills

to degree programs?

- How does the performance of the baseline approach compare to the final graph enrichment process?
- What are the specific limitations and challenges in matching fresh graduates' skills with degree programs based on university curricula and course descriptions?

In this thesis project, I aim to address several problems that hinder the effective utilization of university curricula and course descriptions to improve the quality of degree-skill relationships for fresh graduates. The first problem involves the challenge of collecting and processing data from diverse university course catalogues due to variations in format and structure. To overcome this, I will explore meticulous preprocessing and extraction techniques. The second problem relates to the time and resource restrictions that make manual data collection and hand annotations impractical. I propose investigating alternative approaches, such as semi-automated labelling, to overcome these limitations. The third problem revolves around evaluating the quality and coherence of the graph and its effectiveness in matching skills to degree programs. To address this, I will develop evaluation methods and compare the performance of the enriched graph with ground truth data. Finally, the lack of directly comparable related work in the highly specific domain poses a challenge in assessing the project's results. To overcome this, I will conduct a thorough literature review and compare our proposed solution with existing domain-specific studies and methodologies. By implementing these solutions and testing them through various steps, this research project seeks to enhance candidate matching, contribute to talent management practices, and make significant contributions to the fields of HRM and NLP.

1.2.2 Proposed solution

The pipeline will have two steps shown in Figure 1.1- (a) Data collection and Annotation, and (b) Graph Building/Enrichment.

The data used for this is text scrapped from university course catalogues, which give a brief introduction to the program and its learning objectives, and then a small description of each of the courses taught in the program. This data will then be used to enrich the graph with the new node entity.

```
"Particular attention will be paid to the methodologies for using machine learning in NLP   research."
"     O         O      O    O   O   O  O         O         O    O   B-skill  I-Skill O B-Skill I-Skill."
```

Annotations will be done semi-automatically. The graph enrichment section of the pipeline will use similarity measures that would find and associate the most common skills to the degree program name. Based on semantic similarities(word embeddings/ TFIDF score clustering K-means), the degree names will also be consolidated into a tree structure like the one mentioned above with concept-level degree names, associated with all the synonyms of it. The graph building/ enrichment part of the system will be evaluated using subtree similarity scores, where the ground truth of the graph is hand created on the test and dev set.

The methodology of this research will help facilitate further research and innovation in talent management and workforce development. Utilizing course descriptions from university curricula, we are able to capture detailed information about the skills and knowledge associated with different degree programs. This is a targeted approach to facilitating a smoother transition from education to employment. It also serves as

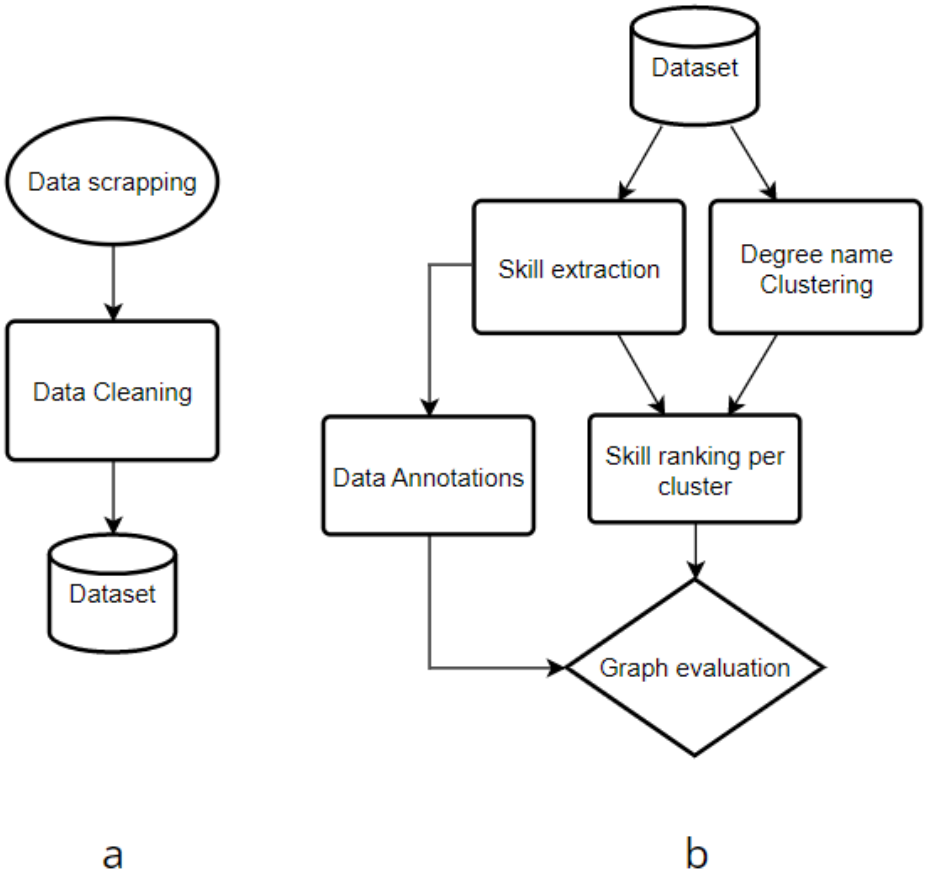


Figure 1.1: Solution flow of this Thesis Project

a valuable resource for in-depth analysis and exploration. Researchers can leverage this graph to gain insights into the evolving landscape of skills for professions, and educational their trends. The methodology opens up avenues for investigating new approaches to talent management, curriculum design, and lifelong learning initiatives, fostering innovation in the field.

Chapter 2

Related Works

The related work that is currently available in the information ranking domain is abundant and I will explore some of the measures that are the most suitable for my task of graph augmentation.

2.1 The Task

The task at hand involves enriching a graph by establishing relations between the existing nodes and a new type of node. The skill extraction service and data collection section will address the acquisition of sources and targets for the graph's relationships. The objective of this thesis is to develop a framework that enhances the existing graph structure by inferring missing relationships between nodes. By utilizing a skill extraction service and carefully curated data collection, the aim is to leverage external sources of information to enrich the graph, thereby improving its completeness and connectivity.

Methodology:

1. **Skill Extraction Service Integration**: Integrate a pre-existing skill extraction service capable of identifying skills and extracting relevant information from unstructured data sources. This integration will provide a comprehensive set of normalized skill concepts, which along with their associated degree names can potentially be used to enrich the graph.

2. **Data Collection**: Design and implement a data collection process to gather additional information, like the category of the skill or which degree/course it is mentioned in, etc, that complements the existing graph.

3. **Relationship Inference**: Apply algorithms or models that can effectively infer relationships in the graph based on the extracted information from the skill service and the collected data. This process should consider the existing graph structure and leverage statistical or machine-learning techniques to make accurate predictions.

4. **Enrichment and Integration**: Integrate the inferred connections between the skill and education nodes into the existing graph structure, ensuring consistency and compatibility with the graph schema. Develop mechanisms to handle conflicts, duplicate information, or noisy data during the enrichment process.

Relationship inference plays a critical role in the graph enrichment task, requiring algorithms or models to predict missing relationships based on information from the skill extraction service and collected data.

2.2 First Approaches

The initial approach involved examining the co-occurrence and frequency of skills within the text. This served as the baseline. Other statistical and probabilistic models are also used in relationship inference. These models incorporate factors such as co-occurrence statistics, similarity measures, or latent variables to determine the likelihood of a relationship between nodes. They handle uncertainty and provide insights into the probabilistic nature of relationships.

Co-occurrence Analysis: This method measures the frequency or likelihood of two entities appearing together in available data sources. Relationships are inferred based on the assumption that entities that frequently co-occur are more likely to have a relationship. Heist (2018) presents in his paper a three-phased approach for extracting co-occurrence patterns and explores their generality when applied to the Document Web. The research questions posed include the possibility of discovering entity co-occurrence patterns locally and globally, the grouping of co-occurrence patterns into different types, and the generalization of these patterns for application to arbitrary web documents.

2.3 Other Statistical Measures

TF-IDF (Term Frequency-Inverse Document Frequency): TF-IDF is a weighting scheme commonly used in information retrieval and text mining tasks. It assigns weights to terms based on their frequency within a document and their inverse frequency across all documents in a corpus. TF-IDF can be used to compute the similarity between documents or text snippets based on shared terms and their importance.

Many teams have explored TFIDF as a scoring mechanism to rank terms for knowledge graph generation. Kim and Chung's team Hyun-Jin Kim (2020) used such a scoring to remove terms with low scores, thus optimizing the KG building process. They created a knowledge base of traffic accidents and safety, for analysis and prediction of emerging risks. By extracting significant knowledge through association rules and generating a graph, the knowledge graph serves as a valuable resource for understanding and organizing information related to traffic accidents and safety. The background of the study is focused on the need to find significant information in massive data generated in real time. With the increasing volume of data, it becomes crucial to improve the speed and usefulness of the association rule algorithm. The conventional association rule algorithms often include words with low importance, leading to low information offering efficiency. Therefore, the study proposes a method to optimize the associative knowledge graph using TF-IDF based ranking scores. By removing words with low importance and creating a knowledge graph based on TF-IDF weights, the study aims to enhance the extraction of significant information. Their results showed that the models they used performed better when the TDIDF score was used for the graph pruning. It ran faster by 22 seconds and also improved the confidence score by 0.01.

In my thesis, I can use the TF-IDF weighting method to assign weights to educational skills or concepts in the university curricula based on their relevance and significance. This can help me identify the most important skills or concepts in the curricula and their relationships with other skills or concepts. I can preprocess the unstructured university curricula data by extracting the educational skills or concepts and calculating their TF-IDF scores. Then, I can use these scores to create a weighted

graph structure, where the nodes represent the educational skills and degree concepts and the edges represent the relationships between them.

TextRank: TextRank Rada Mihalcea (2004) is a graph-based algorithm primarily used for keyword extraction and text summarization. It can be adapted for entity ranking, considering connectivity and relationships within the graph. TextRank applies a ranking algorithm to assign importance scores to entities, identifying the most significant entities within the text. TextRank assumes that the importance of an entity can be determined by its connectivity within the text. Entities that are connected to other important entities or frequently co-occur with them are considered more important. This assumption is inspired by the idea that important entities are likely to be referenced or discussed in relation to other important entities.

Inspired by PageRank pag (2023), TextRank constructs a graph representation of the text, where each vertex represents a unit of text (such as a word or sentence), and edges signify the relationships between these units. The strength of the relationships is determined by the similarity of their contextual information. Through an iterative process, TextRank assigns a score to each vertex in the graph, reflecting its significance within the entire text. This scoring mechanism considers the vertex's local context and recursively incorporates information from the entire text. The score of a vertex is computed recursively, taking into account the scores of the connected vertices. Building on this algorithm's original use case, Liu (2009) , employed the subject model in conjunction with the PageRank algorithm to extract keywords based on word importance, considering the significance of words within the subject's context. The proposed method for keyphrase extraction improves upon existing graph-based ranking methods by outperforming them in F1 measures by 9.5%. This and the following strengths of the algorithm have proven it to be a good candidate for the task at hand.

TextRank is an unsupervised algorithm, which means it does not require pre-labelled data or training. This makes it versatile and applicable in scenarios where labelled data for entity ranking may be limited or unavailable. By relying on the inherent structure and relationships within the text, TextRank can perform entity ranking without the need for extensive supervision or training. TextRank is also not limited to specific domains and can be applied to various types of texts and languages. This flexibility allows it to handle entity ranking tasks in different domains, such as news articles, scientific papers, or social media posts. By adapting the input text and incorporating domain-specific knowledge or heuristics, TextRank can be tailored to specific domains and achieve effective entity ranking.

2.4 Advanced Methods with LLMs

The ProP (Prompting as Probing) system Alivanistos (2022), implemented for the "Knowledge Base Construction from Pre-trained Language Models" challenge, proved useful for knowledge base construction in their example. Alivanistos and his team employ GPT-3, a large Language Model, for Knowledge Base Construction (KBC) using a technique known as "Prompting as Probing." This approach involves generating prompts that elicit specific responses from the language model, thereby constructing a knowledge base. ProP combines various prompting techniques and post-processing methods to enhance the accuracy of GPT-3's predictions for KBC. ProP focuses on predicting possible objects of a triple given the subject and relation, producing sets of objects for different relation types. They tried various prompting styles, and their

results showed that triple-based prompts worked better than natural language prompts. The authors' evaluation study on ProP revealed what is essential:

- Manual Prompt Curation: Manually curating prompts involves carefully crafting and refining them to elicit the desired information from the LM. This process helps in optimizing the LM's performance by providing specific instructions and constraints for generating responses.
- Variable answer sets: The authors found that it is crucial to encourage the language model to provide answer sets of varying lengths, including empty answer sets. This approach improved the quality of the final predictions.
- True/false questions: The study showed that incorporating true/false questions as prompts can increase precision in the suggestions generated by the language model.
- Size of the language model: The authors found that the size of the language model used in ProP is a critical factor in achieving better performance.

These techniques significantly improved performance, with ProP outperforming the baseline by 36.4 percentage points. While my thesis focuses on a different domain, I drew inspiration from ProP's prompting techniques/results, and adapt them to my research context.

Chapter 3

Data and Annotation study

The dataset used in this thesis is built by collecting text from university course catalogues, which give a brief introduction to the program A.1 and its learning objectives, and then a small description of each of the courses taught in the program A.3. This data will then be used to extract skills learned in that degree. This will be normalized over many universities offering the same/ similar degrees. Creating a "core concept" tree graph-type structure of degree names, and associating skills to each core degree. This will then be added as a new node entity in the knowledge graph, making it an education-skill-profession KG. University course descriptions were selected as the primary data source due to the following merits:

- **Comprehensive Coverage**: University course descriptions provide detailed information about the curriculum, courses, and learning outcomes of degree programs. They offer a holistic view of the skills students are expected to acquire throughout their studies.
- **Structured Information**: Course descriptions are typically well-organized and standardized within universities. They contain explicit details about topics, modules, and learning objectives, facilitating the extraction of relevant data for analysis.
- **Alignment with Academic Standards**: University programs adhere to established academic standards and guidelines. This alignment ensures that the skills mentioned in the course descriptions are relevant and recognized within the respective field or industry.
- **Accessibility**: University course descriptions are often publicly available on institution websites, enabling easy access for research purposes. This accessibility allows for the analysis of a wide range of degree programs across disciplines.

It is also essential to recognize the limitations of relying solely on university course descriptions:

- **Limited Scope**: University websites may not provide comprehensive or up-to-date information about the skills associated with specific degree programs. The focus is often on general descriptions rather than an extensive list of skills.
- **Bias and Subjectivity**: University program descriptions may reflect the institution's perspective and priorities, which might not align with broader industry

or job market requirements. The information provided can be subjective and biased towards promoting their programs.

For this study, data was collected from 12 universities across Europe, focusing on master's and bachelor's degree programs. These degree levels were selected as they represent the largest section of degrees in the client's database. The 12 universities A.1 chosen for data collection were among the top 5 universities in their respective countries and had static web pages that could be scraped for course descriptions. The data was restricted to the English language. By combining multiple data sources, a more accurate and representative mapping of skills to education degree names can be achieved, enhancing the validity and reliability of the research findings.

3.1 Data collection

The data for this study was collected through web scraping techniques, specifically using a Python script developed for this purpose. Web scraping involves extracting information from websites with reusable automated code, and it provides a means to gather large amounts of data efficiently.

The Python script used for web scraping was designed to target university websites and retrieve information about their programs, courses, and descriptions. The script utilized the 'requests' library req to send HTTP requests to the target websites and retrieve the HTML content of the pages. The 're' library re was employed for applying regular expressions (regex) to parse and extract specific pieces(like the title of the page that represented the degree/course names or learning objectives paragraph from the course descriptions etc.) of information from the HTML.

The data collection process involved creating a list of dictionaries, named 'dict_of_domains', where each dictionary represented a university search page that needed to be scraped. Each dictionary in 'dict_of_domains' contained custom regex patterns tailored to the structure of the respective university's website 3.1. These patterns were designed to capture relevant information such as program titles, URLs of degree programs, course URLs, specialization track URLs, program and track descriptions, course descriptions, and course names.

```
dict_of_domains=[{"Title":"<title>(.+?)</title>", #regex for pulling the title of the page, which is the title of the program generally
                  "SearchPage":"https://en.uit.no/education", # course catalogue url
                  "Degreeregex":'tittel"><h2><a href="(.+?)"', # regex to pull individual degree urls
                 "Courseregex":'<a href="(https://(?:en.|)uit.no/utdanning/.*?/emne.+?)">', #regex  to pull individual courses URL in a program URL
                 "tracksregex":'<a href="(https://(?:en.|)uit.no/education/program.+?)">', ##regex  to pull individual specialization tracks URL in a program URL
                 "prog_desc":['(?s)<span>.*?Learning outcomes.*?</span>(.+?)<span>','(?s)<span>.*?Program description.*?</span>(.+?)<span>'],#regex  to pull broad program
                 "course_desc":'(?s)Course content.*?(.+?)Language',#regex  to pull individual course description in a program
                 'CourseName':'<meta.*?description.*?content=(.+?)/>'}] #regex  to pull individual course name/title in a program
```

Figure 3.1: Sample of one such dictionary regex

The Python script iterated over each dictionary in 'dict_of_domains', making HTTP requests to the search page URL specified in the dictionary. The HTML content of the page was retrieved and subsequently processed using the regex patterns defined in the dictionary. The extracted information was then stored in a nested dictionary structure, representing the different levels of data hierarchy, including degree programs, specialization tracks, and individual courses. This data was accumulated in a list named 'data'.

To ensure comprehensive data collection, the script was designed to handle variations in webpage structures across different universities. For instance, if a university had multiple pages in its course catalogue, each page was added as a separate dictionary in 'dict_of_domains', allowing the script to navigate through multiple pages and collect all the available data. Upon completion of the scraping process, the 'data' list containing the extracted information was saved to a JSON file. A sample of the file is included in the GitHub repository of this thesis Git.

3.1.1 Data cleaning and preprocessing

Data cleaning is a crucial step in the data preprocessing pipeline, as it ensures the quality and consistency of the collected data. In this study, the collected data from university websites underwent a cleaning process using a Python script. The script utilized the 'cleantext' library cle and implemented custom cleaning functions to transform the raw text into a cleaner and more standardized format.

The cleaning process involved several steps to address various issues commonly found in web-scraped data. The 'clean_data' function was developed to handle these steps and perform the necessary transformations. The steps performed in the cleaning process were as follows:

- **Removal of Text in Brackets**: Any text enclosed in brackets, such as parentheses, square brackets, or curly braces, was removed. This step aimed to eliminate any unnecessary information or annotations present in the text, especially since some HTML got picked up in the data as noise.

- **Removal of Special Characters**: Special characters, such as semicolons, were removed from the text.

- **Fixing Unicode Characters**: In some cases, the scraped text contained Unicode characters that could cause issues during subsequent processing or analysis. The cleaning process addressed this by fixing any Unicode character problems and converting them to their ASCII representation.

- **Conversion to ASCII Coding**: To ensure compatibility and uniformity, the text was converted to ASCII coding. This step involved transforming any non-ASCII characters into their closest ASCII equivalents.

- **Cleaning URLs, Emails, and Phone Numbers**: The script removed URLs, email addresses, and phone numbers from the text. These pieces of information were considered irrelevant to the analysis and could potentially introduce noise or privacy concerns.

- **Handling Line Breaks**: Line breaks within the text were removed to ensure a coherent and consistent format for further analysis.

The 'clean_data' function was applied to the scraped data from all the universities, which were stored in JSON files in the "ScrappedData" directory. The cleaned data was then converted into a CSV file for further analysis. Each course within a program was treated as an individual entry in the CSV, providing a structured and organized format for subsequent processing.

3.2 Entity extraction

The cleaned data obtained from the university websites, was further processed using entity extraction techniques. Entity extraction automatically identifies named entities to provide structured information from unstructured text for further use. In my case, the entity type is skill_name. I used the SkillService provided by Textkernel for this, which is based on the proprietary technology of string matching and word disambiguation models. However, skill extraction services in general are designed to identify and extract skills or competencies mentioned in text, such as resumes, job descriptions, or social media profiles. The extracted skills can vary depending on the application domain or industry. For example, in the context of job matching or recruitment, skill extraction services can help match job seekers with suitable job openings by automatically identifying relevant skills from their resumes or profiles. In talent management or workforce planning, these services can assist in identifying skill gaps or conducting skill inventories for employees.

The exact capabilities and features of a specific skill extraction service can vary. The service I used extracted skills based on a list of predefined skills the company has. It is a large graph with 4 categories at the top - 'IT skills', 'Professional Skills', 'Language' and 'Soft skills'. Each of these has various subcategories in them, and those in turn have skill concepts listed under them. Concepts of skills serve as a foundational framework and a standardized representation of skills, enabling the categorization, comparison, and evaluation across many categories. Defining what a skill is very crucial for capturing and utilizing skill-related information, and this definition has been hand curated by the team of Linguists at Textkernel. Each skill concept has various surface forms/ synonyms or multilingual translations associated with them, to further increase the robustness and coverage of the system. The Textkernel Skills Classification includes over 12,000 concepts that categorize over 315,000 synonyms in 15 languages(although this thesis is only focused on English data). As per the company's requirements, I stuck to the closed-world list of skills. Given how well-maintained and built their skill service was, it was the clear choice in skill extraction services.

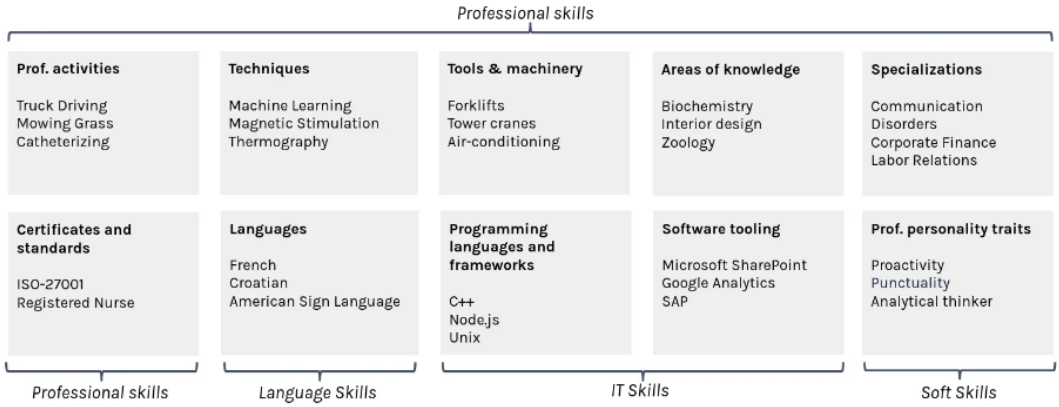


Figure 3.2: Skill Service TextKernel

Free text is provided as input through an API endpoint to the skill service. It processes the text and generates a JSON file containing the location of the surface forms of detected skills, the corresponding surface form of the skill, the normalized skill concept,

and the skill category (e.g., soft skill, IT skill). However, it was observed that the data obtained from the skill service contained some noise and inaccuracies. For instance, skills like teaching were frequently mentioned across various degree programs due to the noisy data collection in the text-scraping section of the thesis. they captured information regarding teaching methods in each course thereby skewing the skills extracted. To address this issue, a rule-based approach was employed to remove such skills and ensure data cleanliness. The thesis was also limited to IT skills and Professional skills, narrowing down the scope of analysis to these particular categories.

3.3 Statistics About the Data

This section presents several key statistics and visualizations that provide insights into the characteristics and distribution of the data used in the study. The statistics are derived from the dataset that includes information about skills and their association with degree programs. The visualizations help visualize the patterns and trends within the data.

In this thesis, the statistical analysis of a dataset consisting of 17,593 records has been conducted. The data represents various attributes related to educational courses, including DegreeURL, DegreeName, ProgDesc (course description) and desc. The total data size is 17,593 entries. Each entry is a course in a degree.

The examination of missing values revealed that the "ProgDesc" column has the highest number of missing values, with 34.08% of the data being absent. No duplicate records were found in the dataset, ensuring data integrity and uniqueness. The "DegreeURL" column contains 2,067 unique values, whereas the "DegreeName" column has 2,037 unique values. The "ProgDesc" column, which represents the course descriptions, exhibits 1,927 unique values.

The statistics related to course description length are as follows: there are 11,598 non-null entries, with a mean description length of 7,193 characters. The course descriptions vary widely, from 6 characters to 197,178 characters. The descriptions' length distribution shows that 25% of the descriptions have a length of 2,033 characters or less, while 75% have a length of 9,633 characters or less.

The word frequency analysis of the course descriptions revealed the most common words and phrases. Some prominent terms include "university," "studies," "Lund," "students," and "programme," indicating the recurring themes and topics within the courses. Additionally, terms like "tuition," "English," and "application" highlight essential aspects of the educational programs.

3.3.1 Category Distribution of Skill Types

Figure 3.3 shows the category distribution of skill types. This bar chart illustrates the relative frequencies of different skill categories present in the dataset. It provides an overview of the skill domain coverage within the educational context. The unbalanced distribution towards professional skills compared to IT skills in the category distribution of skill types reflects the emphasis placed on developing general professional competencies within the analyzed degree programs. This suggests a focus on preparing students for a wide range of professional roles and industries, while IT-specific skills may be integrated into broader skill sets or addressed through specialized IT programs.

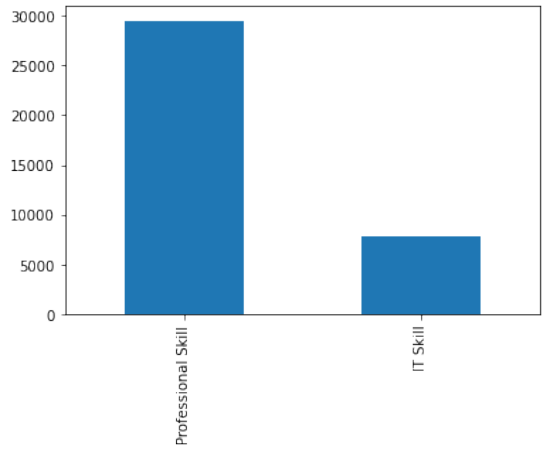


Figure 3.3: Catgeory Distribution of skill types

3.3.2 Distribution of Degree Level

Figure 3.4 displays the distribution of degree levels. It shows the proportions of different degree levels, between bachelor's and master's present in the dataset. This information helps understand the representation and focus of different degree levels in relation to the skills being taught. the dataset is unbalanced towards Master's degrees. this is probably due to the variety of specializations available in Master's degrees.

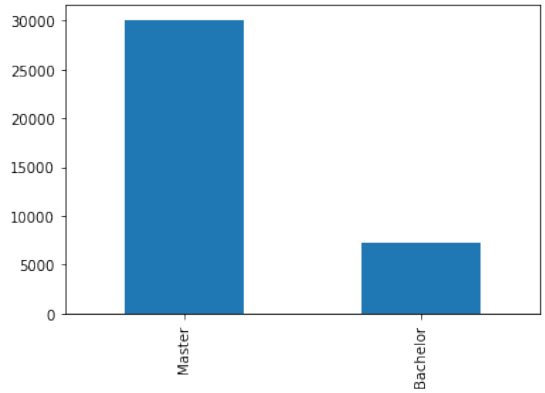


Figure 3.4: Distribution of Degree Level

3.3.3 Top Most Frequent Skills after Cleaning

Figure 3.5 showcases the top most frequent skills after the cleaning process. It presents a bar chart of the skills ranked by their frequency in descending order. This visualization offers insights into the skills that are most commonly taught across the degree programs considered in the study. The prominence of STEM skills in the topmost frequent list highlights the alignment of educational programs with industry demands and job market trends. It indicates that the analyzed degree programs strive to equip students with the foundational skills necessary to succeed in STEM-related professions.

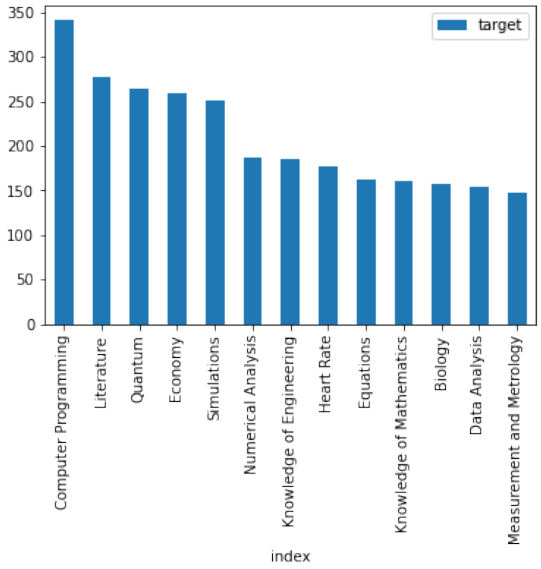


Figure 3.5: Top most frequent skills after cleaning

However, it is important to note that the prevalence of STEM skills in the topmost frequent list may be influenced by various factors, including the specific dataset used, the composition of the degree programs considered, and the prevailing educational and industry contexts. Further analysis and examination of additional datasets would provide a more comprehensive understanding of the skill distribution across different disciplines.

3.3.4 Bucketed Version of Distribution of How Many Degrees Have How Many Skills

Figure 3.6 displays a bucketed version of the distribution of how many degrees have how many skills based on co-occurrence alone. This bar chart provides insights into the number of degrees associated with different skill quantities. It offers an overview of the skill composition across the degree programs and highlights the commonalities and variations in skill requirements. here the x-axis shows the number of skills found in a degree. the y-axis shows how many degrees with that many skills associated with it on an average at baseline. The x-axis has been bucketed with an interval of 10, for easier visualization. The distribution is a bell curve with a right skew, with a few outliers, suggesting that the majority of degrees have a moderate number of skills associated with them. This indicates a typical pattern where most degrees cover a diverse range of skills but do not excessively focus on a large number of skills.

The distribution reveals that there is a concentration of degrees with a moderate number of skills, as indicated by the peak of the curve. This suggests that the majority of degree programs aim to provide a balanced set of skills relevant to their respective fields of study. Such a distribution aligns with the concept of a well-rounded education, where programs aim to equip students with a comprehensive skill set.

The outliers in the distribution represent degrees that require a significantly higher number of skills compared to the majority of programs. These outliers could indicate

specialized or niche programs that focus on highly specific areas of study and demand a broader range of skills. Identifying and analyzing these outliers can provide valuable insights into unique educational offerings and their associated skill requirements.

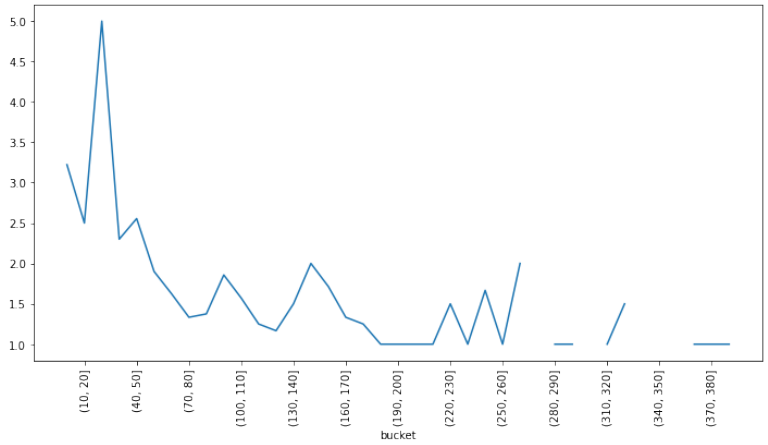


Figure 3.6: Bucketed version of Distribution of how many degrees have how many skills based on co-occurrence alone

These statistics and visualizations offer valuable insights into the dataset, including the distribution of skill categories, degree levels, and skill quantities within degree clusters. They provide a foundation for further analysis and exploration of the relationships between skills and degree programs.

3.4 Evaluation of Data Quality

In this section, I evaluate the data quality of the collected dataset from university course catalogues for mapping skills to education degree names in the field of NLP. The evaluation focuses on the attributes of richness and detail, as well as specificity and granularity, which are crucial for a thorough understanding of the skills taught within each degree program.

The manual examination of course descriptions revealed a generally satisfactory level of detail provided in the dataset. The descriptions contained comprehensive information about the topics covered, methodologies used, and learning outcomes associated with each course. This indicated that the dataset had a good level of richness and detail, enabling a thorough understanding of the skills taught within each degree program.

The analysis of skill frequency 3.6 indicated a dataset with a relatively high level of specificity and granularity. The frequency of explicit skill references within the course descriptions was notable, suggesting that the dataset captured the nuanced skills taught in the field of NLP. This specificity and granularity are crucial for accurately mapping skills to education degree names and ensuring a comprehensive representation of the skills acquired through these programs.

The skill variety 5.3 revealed a wide range and diversity of skills mentioned in the dataset. This indicated a comprehensive understanding of the skills associated with different courses and degree programs within the NLP domain. The dataset exhibited a balanced coverage of various skills, reflecting the multidimensional nature of the field

and providing a holistic perspective on the skills taught.

Lastly, the evaluation of the depth of skill descriptions within the dataset 3.5 indicated a favourable level of granularity and specificity. The skill descriptions went beyond generic terms and often provided specific examples, methodologies, or applications associated with each skill. This level of detail contributed to a more accurate representation of the skills taught within the degree programs, enhancing the quality and usefulness of the dataset.

The evaluations conducted on the collected dataset revealed positive findings regarding its quality. The dataset demonstrated a good level of richness and detail in course descriptions, specificity and granularity in skill mentions, a wide variety of skills, and alignment with learning outcomes. These findings affirmed the comprehensiveness and relevance of the dataset for mapping skills to education degree names in the field of NLP, providing a strong foundation for further analysis and exploration.

3.5 Data Annotations

Data annotations play a vital role in machine learning tasks, as they provide labelled examples that enable the training and evaluation of models. In this study, data annotations were performed to establish a connection between skills and university degree programs. This section discusses the annotation process, the rationale behind the chosen annotation type, the annotated data section, the annotator, and the guidelines used.

3.5.1 Annotation Process

The annotation task involved indicating whether a particular skill is typically taught in a specific degree program. For each skill-degree connection present in the data, the annotator assigned a binary annotation. A value of '1' indicated that the degree program typically teaches that skill, while a value of '0' indicated that it does not. The binary annotation type was chosen to simplify the annotation task and make it more manageable. It allows for a straightforward decision for each skill-degree connection, minimizing ambiguity and ensuring consistency. Pros of the binary annotation type include its simplicity, clear decision-making process, and ease of interpretation. Cons may include potential subjectivity in determining the typicality of a skill within a degree program. However, this was mitigated by providing annotation guidelines A.1 and leveraging the expertise of the annotator.

3.5.2 Annotated Data

For the annotation task, a subset of the data was selected to represent a diverse range of degree programs. Specifically, 40-degree clusters were handpicked from the baseline attempt. These clusters were chosen to ensure diversity across topics and contained 50 or more associated skills. The annotated data was split into a development set and a test set. The development set had a maximum of 50 skills annotated per degree cluster, while all skills were annotated in the test set. Importantly, the development and test sets had no overlap to ensure unbiased evaluation.

The annotations were conducted by a linguist within the company and me. The annotator had a strong understanding of language and expertise in the domain being

analyzed. Each degree cluster annotation took approximately 6-10 minutes to complete. Annotation guidelines were established to ensure consistency and quality in the annotations. These guidelines A.1 provided instructions on the criteria for determining whether a skill is typically taught in a degree program, taking into account the curriculum and educational goals associated with each program.

3.6 Inter-Annotator Agreement

This section focuses on the assessment of inter-annotator agreement (IAA) between the researcher and a linguist annotator during the process of annotating skills and their associations with education degree names in the field of NLP. IAA provides insights into the level of agreement between annotators and indicates the consistency and reliability of the annotations. The calculation of IAA involved comparing the annotations independently made by me and the linguist annotator on a subset of the dataset. Both annotators followed the established annotation guidelines discussed in the previous section. The agreement was measured using Cohen's Kappa coh (2023).

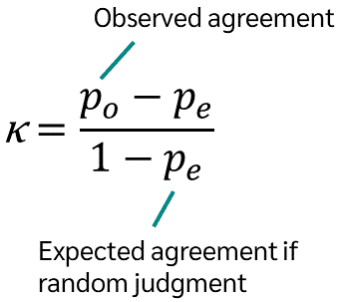


Figure 3.7: Cohen's Kappa Dat

The resulting IAA score was found to be 0.74. This indicates a moderate level of agreement between me and the linguist annotator. While a score of 0.74 demonstrates a substantial level of agreement beyond chance, there is still room for improvement to achieve a higher degree of consistency in the annotations. It is important to note that achieving a higher IAA score may require ongoing collaboration, continuous training, and refinement of the annotation process. The complexities of mapping skills to education degree names in the field of NLP, coupled with potential ambiguities in the data, can contribute to disagreements between annotators. However, the moderate IAA score of 0.74 indicates a solid foundation for further analysis and research.

Chapter 4

Theoretical Framework

This chapter has the collection of concepts, theories, and principles that will provide a foundation for understanding the theoretical framework of the thesis.

4.1 Clustering

Clustering clu (2023) is a technique used in data analysis and machine learning to group similar data points or objects together based on their inherent characteristics or similarities. The goal of clustering is to identify patterns, structures, or natural groupings in a dataset without any prior knowledge or predefined classes.

In clustering, a dataset consists of multiple data points, each described by a set of features or attributes. The clustering algorithm analyzes the data and assigns each data point to a specific cluster, such that data points within the same cluster are more similar to each other compared to data points in different clusters. The similarity or dissimilarity between data points is typically measured using a distance metric or similarity measure.

Clustering algorithms aim to optimize the intra-cluster similarity (similarity within a cluster) and inter-cluster dissimilarity (difference between clusters). The choice of clustering algorithm depends on the nature of the data, the desired clustering structure, and the computational requirements.

This section delves into the application of textual clustering techniques for the purpose of clustering degree names. Degree names encompass a broad spectrum of variations. This presents a challenge in terms of categorization and understanding the interrelationships between different degree programs. Textual clustering, leveraging semantic and syntactic similarities, emerges as a valuable approach to automatically group similar degree names. This section explores some of the clustering methods, highlighting the advantages associated with them.

4.1.1 K-means

Among the various clustering methods, the widely utilized K-means clustering algorithm kme (2023), an unsupervised machine learning technique, stands out. The name "K-means" derives from the algorithm's objective of partitioning the data into K clusters, where K is a user-defined parameter. This algorithm aims to minimize the within-cluster sum of squares, also known as inertia. It accomplishes this through an iterative

process that entails assigning data points to the cluster with the closest centroid and updating the centroids based on the assigned points.

The K-means clustering algorithm operates as follows:

Initialization:

- Determine the desired number of clusters, K, to be identified within the dataset.
- Randomly initialize K centroids, each representing the center of a cluster. Centroids can be selected at random from the data points or initialized using alternative techniques.

Assignment Step:

- Calculate the distance (e.g., Euclidean distance) from each data point to every centroid.
- Assign the data point to the cluster with the nearest centroid. This step results in the formation of K clusters.

Update Step:

- Recalculate the centroids of each cluster by computing the mean of all data points assigned to that cluster.
- The newly computed centroids represent the updated centers of their respective clusters.

Iteration:

- Repeat the assignment and update steps until convergence is achieved or a specified number of iterations is reached.
- Convergence is attained when the assignment of data points to clusters no longer changes or when the change falls below a predetermined threshold.

Final Result:

Upon convergence, the K-means algorithm produces K clusters, each accompanied by its respective centroid. Each data point is assigned to the cluster whose centroid it is closest to.

The selection of K, the number of clusters, is typically determined through domain knowledge, exploratory data analysis, or the application of evaluation metrics such as the elbow method or silhouette score.

K-means clustering exhibits several advantages:

- It demonstrates relative speed and scalability, rendering it suitable for large datasets.
- It is straightforward to implement and interpret.
- It performs well on datasets characterized by distinct and similarly sized clusters.

However, K-means clustering also possesses certain limitations:

- It necessitates the pre-specification of the number of clusters, K.

- The algorithm's outcomes can be influenced by the initial placement of centroids, potentially yielding different clustering results.
- It assumes that clusters are spherical and possess comparable densities, which may not always be the case.

Determining the optimal value of K, the number of clusters represents a critical step within the K-means clustering algorithm. The choice of K relies on the dataset's characteristics and the specific problem being addressed.

Silhouette Score: The silhouette score sil offers a means to evaluate the quality of clustering by considering both the cohesion within clusters and the separation between clusters. It is a measure of how well each data point fits within its assigned cluster and how distinct it is from other clusters. It ranges from -1 to 1, with higher values indicating well-defined clusters. To identify the optimal K using the silhouette score, follow these steps:

- Execute the K-means algorithm for various K values.
- Calculate the silhouette score for each K.
- Select the K that maximizes the silhouette score.

4.2 Knowledge Graphs

A knowledge graph is a structured representation of knowledge, typically in the form of a graph database. They capture entities, their attributes and the relationships between them. It provides a way to organize and store information in a manner that enables reasoning and sophisticated data analysis. Knowledge graphs are designed to model real-world entities, their properties, and the connections between them in a more explicit and structured manner than traditional databases. The graphs have two main components: nodes and edges.

- **Nodes**: Nodes represent entities or concepts. Each node typically corresponds to a specific object, person, place, or idea. In this case, it is a profession or skill or education.
- **Edges**: Edges represent the relationships between nodes. They connect nodes based on their associations or connections. Each edge in a knowledge graph has a specific label that represents the type of relationship between the connected nodes. In this case, there is only one type of connection/relationship - "is_related_to"

By linking nodes through edges, a knowledge graph captures the relationships and connections between different entities. These relationships can be one-to-one, one-to-many, or many-to-many, depending on the nature of the information being represented.

Knowledge graphs find applications in various domains, including semantic search, question-answering systems, recommendation systems, information retrieval, and knowledge representation in artificial intelligence. They enable more advanced forms of data analysis, such as entity resolution, link prediction, semantic search, and reasoning, by leveraging the rich connections and semantics captured within the graph structure.

Textual knowledge graphs are a specific type of knowledge graph that focuses on capturing and representing textual information and its relationships. Unstructured

text, such as documents, articles, or web pages, contains a wealth of valuable information but is inherently difficult to process and analyze due to its lack of explicit structure. The conversion of unstructured text into a structured graph provides a means to organize and represent this textual information in a structured form, allowing for better information retrieval, semantic reasoning, and knowledge extraction.

Knowledge graphs that capture profession-skill relationships play a crucial role in resume and job vacancy parsing, as well as matching processes. These systems leverage the structured information within knowledge graphs to do efficient matching. Resume or job parsing involves mapping skills mentioned in the document to corresponding nodes in the knowledge graph, allowing for skill standardization and improved accuracy. Skill matching algorithms then assess the compatibility and similarity between candidate skills and job requirements, using the relationships and connections in the knowledge graph. Education information in resumes, such as degrees and courses will contribute to inferring additional skills and improves the matching process by considering a wider range of skills.

4.3 Graph Pruning

Graph pruning is a critical process in dealing with graphs, particularly in large-scale or complex structures. Its objective is to remove unnecessary or less relevant nodes and edges from the graph to have a more focused representation. Graph pruning enhances efficiency and improved computational performance. Smaller graphs require fewer resources, enabling faster analysis and traversal. Graph pruning also improves interpretability by simplifying the graph structure and highlighting the most important relationships. By removing irrelevant or noisy elements, the remaining graph becomes easier to understand and visualize, facilitating data exploration and knowledge representation. It also reduces noise by eliminating erroneous or inconsistent data. By retaining only the most significant nodes and edges, the pruned graph allows for targeted exploration and analysis of specific aspects. When applied to textual graphs, graph pruning utilizes similarity measures of textual nodes to identify and remove similar or less informative nodes. The following presents a general approach to graph pruning using similarity measures of textual nodes:

- **Weightage**: Select an appropriate measure to quantify the nearness between textual nodes. Depending on the specific task requirements, common measures like similarity measures from word embeddings like cosine similarity or Euclidean distance can be employed. We can also use other relationship-weighing techniques based on the task at hand. In this case, I chose TDIDF, Co-occurrence, frequency and TextRank algorithm.

- **Threshold and Pruning decision**: Establish a weight threshold serving as a criterion for pruning. Nodes with scores below this threshold are considered irrelevant and potentially pruned from the graph.

- **Graph Modification**: Implement the pruning decisions by removing the identified nodes from the graph, along with their associated edges. The resulting pruned graph exhibits reduced size and complexity.

4.3.1 TF-IDF

One commonly employed weighting scheme in information retrieval and text mining tasks is TF-IDF (Term Frequency-Inverse Document Frequency). It evaluates the importance of a term (word) in a document within a collection of documents. TF-IDF incorporates both the term's frequency in a document (TF) and its rarity across the entire document collection (IDF). Chen (2021)

$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d}$$
$$IDF(t) = log\frac{N}{1+df}$$
$$TF-IDF(t,d) = TF(t,d) * IDF(t)$$

Figure 4.1: TFIDF formula

The TF component of TF-IDF gauges the local importance of a term within a document. It calculates the frequency of a term in a document and normalizes it by the total number of terms in that document. The concept behind TF is that terms occurring more frequently within a document hold greater importance in representing its content. The IDF component of TF-IDF assesses the global importance of a term throughout the entire document collection. It is determined by taking the logarithm of the inverse of the term's document frequency. The document frequency of a term represents the number of documents in the collection containing the term. IDF emphasizes terms that are relatively rare across the entire collection and assigns them higher weights. The TF-IDF weight for a term in a document is obtained by multiplying the TF and IDF values for that term. Higher TF-IDF weights indicate greater importance of the term within both the document and the collection.

TF-IDF can be leveraged for graph pruning of textual data by utilizing the significance of terms in documents to determine the relevance of nodes within a graph. By employing TF-IDF for graph pruning, nodes that are less informative or frequently occur across documents (thus having low TF-IDF scores) can be pruned. Conversely, nodes with higher importance and more specific or unique characteristics are retained in the pruned graph.

4.3.2 TextRank

Another notable algorithm for textual weighting/ranking is TextRank tex. It is a graph-based ranking algorithm specifically designed for text summarization and keyword extraction, inspired by the PageRank algorithm employed by search engines to rank web pages. TextRank can also be applied to identify the most important and relevant keywords in a document.

The TextRank algorithm operates as follows:

- **Graph Construction**: Represent the text as a graph, with nodes representing textual units (such as sentences or words) and edges denoting relationships between the units. Typically, the graph is constructed using co-occurrence information or syntactic dependencies.

- **Node Weighting**: Assign an initial weight to each node in the graph. In the case of text summarization, the initial weight can be uniformly distributed among the nodes. For keyword extraction/ ranking, the initial weight often relies on the term frequency (TF) or TF-IDF values of the nodes.

- **Iterative Ranking**: Perform iterative ranking updates based on the concept of random walks on the graph. The TextRank algorithm updates the weights of the nodes iteratively, considering the weights of their neighbouring nodes. Each iteration influences the weight of a node based on the weights of its adjacent nodes, akin to how PageRank determines the importance of web pages based on the importance of the pages linking to them.

- **Convergence**: Repeat the ranking updates until the algorithm converges. Convergence criteria can be based on a predetermined number of iterations or when the weights of the nodes stabilize.

- **Importance Ranking**: Upon convergence, the final weights of the nodes reflect their importance or relevance within the graph. Nodes with higher weights are considered more significant in representing the content of the text.

TextRank proves valuable in textual graph pruning as it identifies the central nodes within a graph, which often represent key information or representative elements. By ranking the nodes based on their importance, TextRank assists in pruning the less important or redundant nodes from the graph, resulting in a more concise and informative representation. Concerning textual graph pruning, TextRank can be applied by considering the nodes as textual units (e.g., sentences or words) and pruning the nodes with lower ranks or weights.

4.4 Transformers

Transformers have emerged as a powerful architecture in the field of natural language processing (NLP) and have gained significant attention since the publication of the seminal paper "Attention is All You Need" Vaswani et al. (2023). This section provides an overview of the architecture of transformers and delves into their working principles.

The transformer architecture represents a paradigm shift from traditional recurrent neural networks (RNNs) and convolutional neural networks (CNNs) in sequence modelling tasks. It leverages a fully attention-based mechanism to capture long-range dependencies and effectively process input sequences. The core components of a transformer architecture are as follows:

- **Encoder** - The encoder is responsible for transforming an input sequence into a rich representation suitable for downstream tasks. It consists of a stack of identical layers, where each layer comprises two sub-layers: a multi-head self-attention mechanism and a position-wise fully connected feed-forward neural network.

- **Self-Attention Mechanism** - The self-attention mechanism enables each position in the input sequence to attend to all other positions, allowing the model to capture dependencies between words without relying on sequential processing. It computes the attention weights by comparing the similarity between the target position and all other positions using dot products.

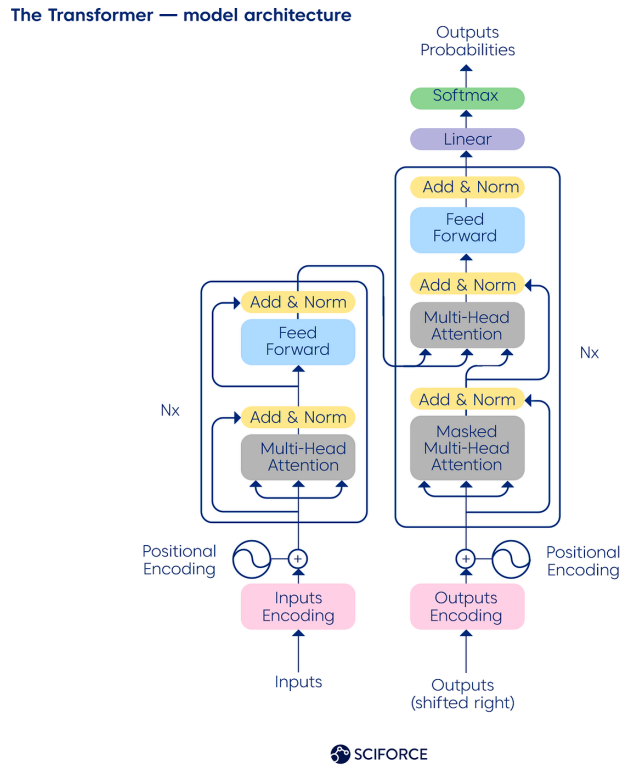


Figure 4.2: Transformer Architecture

- **Position-wise Feed-Forward Network** - After the self-attention mechanism, a position-wise feed-forward network is applied to each position independently. This network consists of two linear transformations followed by a non-linear activation function, such as the rectified linear unit (ReLU).
- **Decoder** - The decoder takes the encoder's output and generates a sequence, autoregressively, one position at a time. Similar to the encoder, it also consists of a stack of identical layers but includes an additional sub-layer: the encoder-decoder attention mechanism.
- **Encoder-Decoder Attention Mechanism** - The encoder-decoder attention mechanism allows the decoder to focus on relevant parts of the encoded input sequence while generating the output. It computes the attention weights by comparing the similarity between the decoder's position and all positions in the encoder's output.

Transformers operate by learning to assign appropriate attention weights to different positions in the input sequence. This attention mechanism enables the model to capture contextual relationships and dependencies effectively, irrespective of their distance in the sequence. The attention weights are determined based on the content of the input sequence, allowing the model to attend to the most relevant information.

During training, transformers employ a technique called "self-attention" to capture dependencies within the input sequence. By attending to all other positions, each position can capture the context of the entire sequence, making transformers highly

effective in understanding long-range dependencies. Additionally, transformers employ residual connections and layer normalization to facilitate gradient flow and stabilize training. At inference time, the decoder generates the output sequence autoregressively by predicting one position at a time based on previously generated positions. This autoregressive process allows transformers to generate sequences of arbitrary length with coherence and fluency.

Transformers have revolutionized the field of NLP by providing a powerful architecture that overcomes the limitations of traditional sequence modelling approaches. With their attention-based mechanism, transformers effectively capture long-range dependencies and enable the generation of high-quality sequences. The success of transformers has led to their widespread adoption in various NLP tasks, establishing them as a key component in modern deep-learning architectures.

4.5 LLM

Rouse (2023) are advanced natural language processing models trained on vast amounts of text data to understand and generate human-like language. They have a deep neural network architecture with multiple layers of attention and transformer-based models. LLMs, such as OpenAI's GPT-3, have been successful in various language-related tasks like text generation, translation, and question-answering.

LLMs are valuable for building knowledge graphs (KGs) because they possess contextual understanding, language coherence, and common sense reasoning abilities. These capabilities help in removing obviously bad results from KGs. LLMs analyze the surrounding text to identify inconsistencies allowing them to filter out incorrect results. They can also detect syntactic and semantic errors in generated text, aiding in the removal of bad results. Additionally, LLMs leverage their knowledge of common reasoning to identify implausible or unlikely information, further helping in eliminating obviously bad results.

For the purposes of this thesis, I used OPenAI's text-davinci-003 model Ope, which falls under the GPT 3.5 series. The GPT-3.5 series models were trained on a blend of text and code data. GPT-3.5 is an advanced language model with enhanced performance and capabilities. It follows a transformer-based architecture, is pre-trained on diverse data, and has a large-scale model size with 175 billion parameters. GPT-3.5 excels in contextual understanding, generating human-like text, and has some limited support for multimodal inputs. It can be fine-tuned for specific tasks and is widely used in natural language processing applications. GPT-3.5 is an improvement on the GPT-3 model. GPT-3 is an autoregressive language model and the third generation in the GPT-n series. It has gained attention for its remarkable language generation abilities, with the largest version, GPT-3 175B, having 175 billion parameters, 96 attention layers, and a 3.2 million batch size. These models have significant potential in various language-related tasks and have been widely adopted in research and business applications.

The architecture of the text-davinci-003 model is not publicly available, but it is an improvement of the GPT-3 architecture Ye and Chen (2023). An overview of the GPT-3 architecture is given below. Sciforce (2021)

Pre-training: GPT-3 is a pre-trained language model, meaning it is initially trained on a large corpus of text from the internet to learn the statistical patterns and linguistic representations present in the data. During pre-training, the model predicts the next

word in a sentence based on the context of the preceding words. This process helps GPT-3 to capture a broad understanding of language and acquire a rich vocabulary.

Fine-tuning: After pre-training, GPT-3 undergoes a process called fine-tuning, where it is further trained on specific downstream tasks or domains. Fine-tuning involves training the model on a smaller dataset that is specific to the task at hand, allowing it to adapt its knowledge and generate task-specific outputs.

Language Generation: The primary strength of GPT-3 lies in its ability to generate human-like text. Given a prompt or a partial sentence, the model can generate a continuation or completion that is contextually relevant and syntactically accurate. GPT-3 achieves this by leveraging the learned patterns from the pre-training stage and fine-tuning on specific tasks.

Zero-shot Learning and Few-shot Learning: One notable aspect of GPT-3 is its zero-shot and few-shot learning capabilities Brown and Mann (2020). Zero-shot learning refers to the ability of the model to perform tasks it has not been explicitly trained on. By providing a task description or an example, GPT-3 can generate reasonable responses without any specific training for that task. Few-shot learning extends this capability by allowing the model to adapt to a new task with just a few training examples.

4.6 LLM prompting

Prompting of language models (LLMs) for knowledge graph completion (KG completion) involves providing specific textual prompts that contain partial information about the desired task. The process of refining prompts and exploring their effects on LLM responses can lead to interesting findings. During experimentation, I observed various phenomena, including the order of "yes" or "no" affecting results and ambiguous prompts generating random outputs. Additionally, the use of synonymous words in prompts has resulted in inconsistent outcomes.

To address these challenges, a combination of prompting styles was adopted. I formatted the prompt to provide context and indicate the desired completion task clearly. Starting with simple prompts that explicitly state the missing information is recommended. These initial prompts had to be concise and unambiguous. If the results were not satisfactory, I added complexity gradually by providing additional context or background information. Experimenting with different prompt variations helped in understanding the model's response. Including example completions in the prompt further guided the model. Demonstrating correct or desirable completions for similar tasks showed that the model learned from the examples and generated more accurate predictions.

The process of iterating and refining prompts is crucial. Evaluating the model's responses and making adjustments based on the observed results are necessary steps. Experimenting with different prompt variations, modifying the prompt structure, and refining the wording led to improved performance. This iterative process was continued until the desired accuracy and quality of completions were achieved.

Chapter 5

Methodology

This chapter contains information about knowledge graph construction/augmentation and the experimental approach that was used for the task. It describes the system setup.

5.1 Degree Name Clustering

5.1.1 Preprocessing

As part of the University data that was collected, the page titles were saved too. These page titles were the names of the degrees/ programs. the unique set of this text is what is referred to as data in this section. Before proceeding with the clustering, data preprocessing was performed to clean and refine the degree names. This involved removing stop words, punctuation marks, and irrelevant information that could potentially hinder the clustering process. Function words, such as articles, prepositions, and pronouns, were eliminated as they do not contribute significantly to the underlying semantic meaning of the degree names. Additionally, the text enclosed within brackets, university names, and domain-specific function words like "bachelor," "year," and "first cycle" were also removed to ensure that the clustering focused solely on the essence of the degree programs. 5.1 shows the before and after cleaning text for a small sample of hand-picked examples from mathematics degrees.

To enable numerical representation of the degree names, word embeddings were employed. I utilized the Spacy library, which provides efficient and effective methods for transforming text into distributed vector representations. These embeddings capture the semantic relationships between words, allowing for meaningful comparison and clustering of the degree names. By converting the degree names into numerical representations, I was able to leverage various machine-learning techniques, including clustering algorithms.

5.1.2 Clustering

For the clustering phase, I employed the popular k-means algorithm kme. This algorithm partitions the dataset into a specified number of clusters, with each degree name being assigned to the nearest cluster centre based on its numerical representation. To incorporate the k-means algorithm into the pipeline, I integrated it into the Spacy framework spa, enabling seamless clustering of the degree names. The results of the clustering were evaluated using the silhouette score, a widely accepted metric for

degree	clean
Master Cycle - Mathematics - master program - ...	Mathematics program
Master Cycle - Applied Mathematics - EPFL	Applied Mathematics
Master in Mathematics and applications - Catal...	Mathematics applications
Mathematics (Master 2022-2023) - Prospectus - ...	Mathematics
Master's Programme in Mathematics	Mathematics
Bachelor Cycle - Mathematics - EPFL	Mathematics

Figure 5.1: Math Cluster

assessing the quality and consistency of clustering. This allowed me to measure the coherence within each cluster and the separation between different clusters, providing insights into the effectiveness of the clustering process.

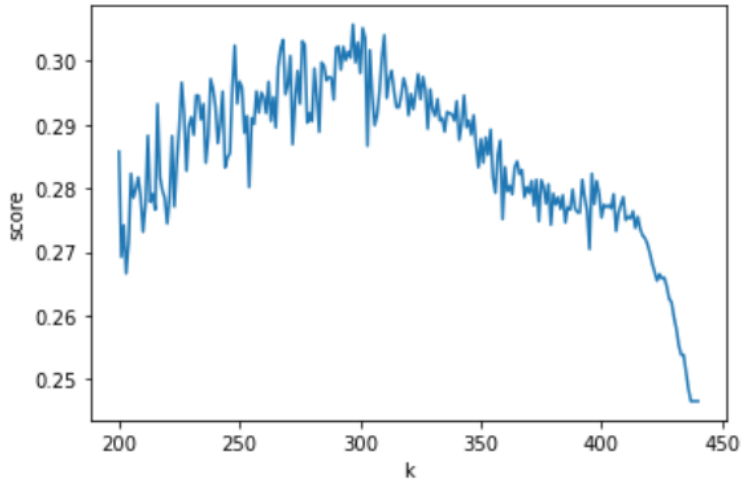


Figure 5.2: Choosing K

Additionally, to determine the optimal number of clusters, I performed a fine-tuning process by conducting a grid search. By systematically varying the number of clusters, I evaluated the impact on the clustering performance shown in 5.2 using the silhouette score. Through this iterative process, I identified that the most suitable number of clusters for our dataset was 290. This fine-tuned parameter ensured that the clustering captured the desired level of granularity and revealed meaningful patterns within the degree names.

5.1.3 Distribution of How Many Skills per Cluster

Figure 5.3 illustrates the distribution of how many skills are associated with each degree cluster. It provides a histogram that showcases the frequency of clusters with a specific number of skills. This information highlights the variation in the number of skills within different degree clusters and gives an indication of their diversity. This is a reverse j curve, with the highest number of skills in a cluster being 1200, and it slowly tapers down from there. The presence of STEM (Science, Technology, Engineering, and Mathematics) degrees as the highest frequency count clusters in the distribution is noteworthy. This suggests that STEM programs typically require a broader range of skills compared to other fields of study. The higher skill counts in STEM degrees may be attributed to the technical nature and interdisciplinary nature of these programs.

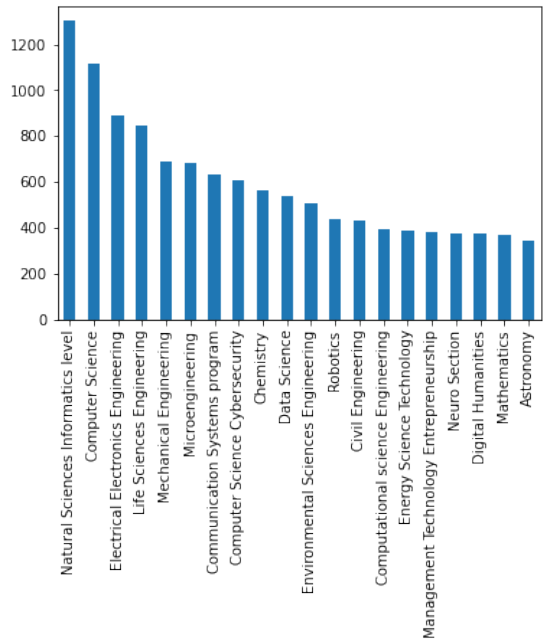


Figure 5.3: Distribution of how many skills per cluster(top20)

5.1.4 Distribution of Unique Degrees per Degree Cluster

Figure 5.4 presents the distribution of unique degrees per degree cluster. It shows the frequency of degree clusters that contain a specific number of unique degrees. This visualization sheds light on the heterogeneity of degree offerings within each cluster. there is not a very considerable amount of overlap between the previous graph and this one, suggesting that the high skill count in the last graph for STEM was not due to inconsistent degree cluster sizes.

5.2 KG

In this section of the methodology chapter, I will provide a comprehensive description of the Knowledge Graph (KG) used in the study. The KG represents a profession-skill

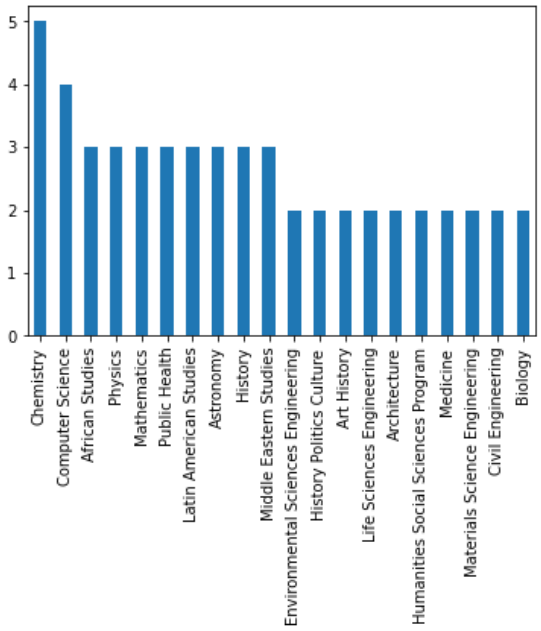


Figure 5.4: Distribution of Unique Degrees per Degree Cluster

graph and serves as a valuable resource for mapping relationships between professions and skills. The existing KG structure has been thoroughly analyzed to identify key nodes and relationships, forming the foundation of the research.

The KG consists of two primary node types: professions and skills. Each node represents a concept-level profession or skill, capturing the broad domain of knowledge associated with them. For example, a profession node could represent a *data scientist*, *software engineer*, or *project manager*, while a skill node could encompass *Python programming*, *data management*, *deployment*, *PyTorch*, and more. These nodes are interconnected with various surface forms or surface realizations, representing specific job titles or variations within the profession. For instance, a *data scientist* profession node may be connected to surface forms such as *principal data scientist* or *data analyst specialist*.

The KG has been meticulously curated by a team of linguists within our company, aligning it with the organization's needs and incorporating user feedback. This manual curation process ensures that the KG remains relevant and accurate. The team of linguists consistently cleans and adds new nodes and relationships to the KG on a monthly basis. The connections within the KG do not possess directional properties.

Each node type in the KG possesses associated attributes and properties, most of which are irrelevant to the scope of this project. Skill nodes have a unique attribute known as the category. In our project, I focused on two specific categories: professional skills and IT skills. This attribute categorizes the skills accordingly, allowing us to limit the scope of our thesis to these specific skill types. I will expand the KG by introducing a third node type at the concept level, focusing on education. This addition will involve incorporating degrees as concept-level nodes and establishing connections between these nodes and their respective surface forms. For example, a *data science degree* node could be connected to surface forms such as *Master of Science in Data*

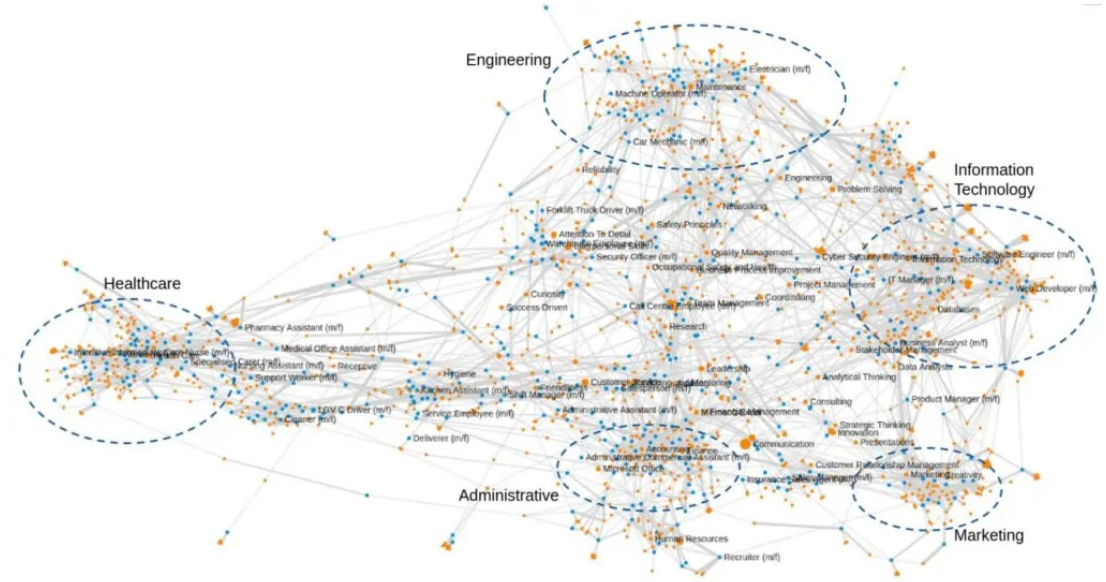


Figure 5.5: Textkernel KG

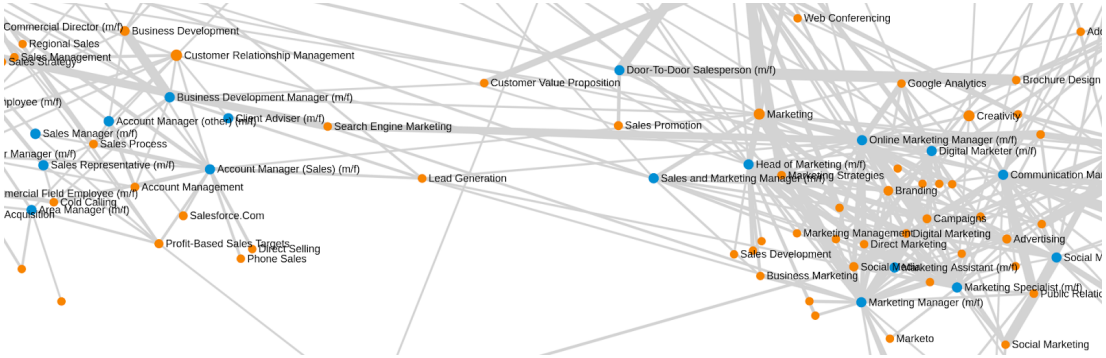


Figure 5.6: Textkernel KG Zoomed

Science or *Bachelor of Technology in Data Analytics*. To ensure coherence and completeness within the KG, the skills already present in the graph will be used to establish appropriate connections to the newly introduced education node. This enhancement will facilitate a comprehensive understanding of the educational background associated with specific professions and skills, providing valuable insights for individuals and organizations alike.

To extend the KG and establish connections between education and skills, I will leverage the degree names obtained during the normalization step mentioned in the data collection chapter. Additionally, I will utilize the associated skills extracted from the text descriptions of the degrees. This approach will enable the identification of relationships between education and skills based on the presence of specific skills being taught as part of each degree program.

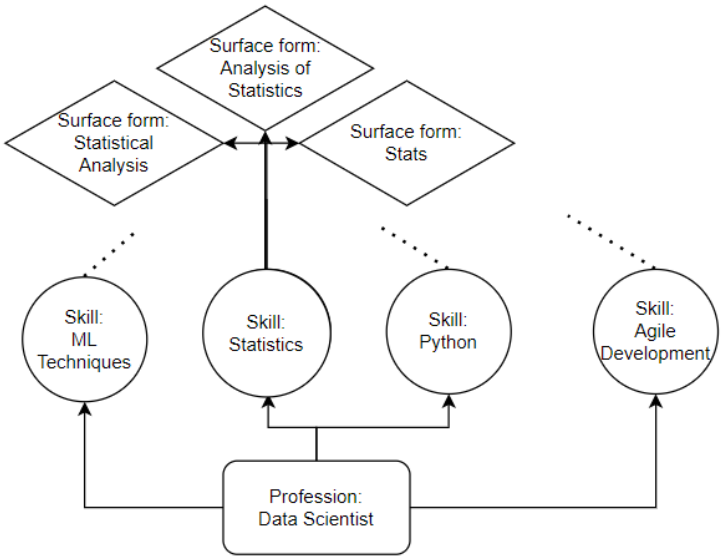


Figure 5.7: Profession-skill KG

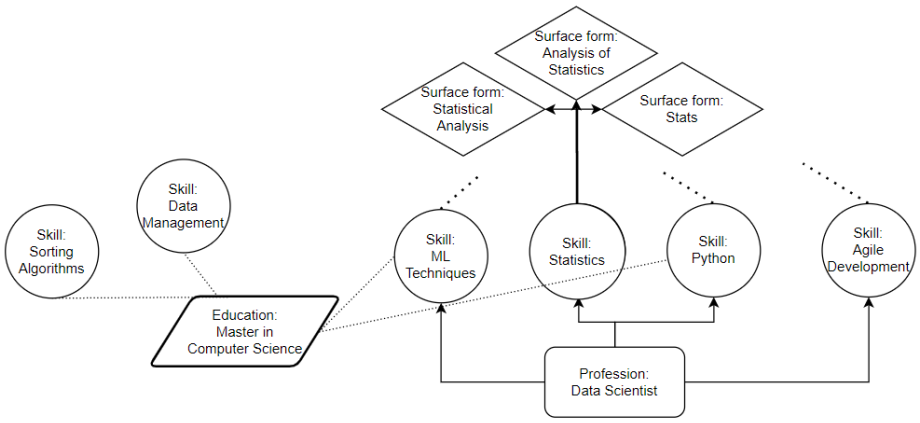


Figure 5.8: Adding Education Node to KG

5.3 Graph Pruning

To determine the existence of a relationship, I will consider whether a particular skill is commonly associated with a specific degree. By analyzing the text descriptions and mapping the extracted skills to the corresponding degrees, I will calculate the average occurrence of each skill within the degree programs. If a skill appears frequently across various degree descriptions, it suggests that the skill is commonly taught as part of those programs. Consequently, a relationship will be established between the education node (representing the degree) and the skill node.

Once the relationships between education and skills are established, I will employ a

skill ranking technique to prioritize the most relevant skills within the KG. This ranking will allow us to identify the top-k skills that hold the most significance in relation to the degrees. By applying the skill ranking methodology, we can effectively prune and focus on the most important skills while filtering out less relevant ones.

5.3.1 Baseline

In order to establish a benchmark for comparison, I employed a frequency baseline approach. This approach involves utilizing the occurrence of skills in the degree descriptions and assigning weights to the skills based on their frequency within the degrees. The higher the frequency of a skill in the degree descriptions, the more important it is considered in the ranking. This frequency baseline serves as a starting point for evaluating the significance of skills in relation to the degrees.

To ensure fair comparison and benchmarking, certain normalization techniques or adjustments were applied to the frequency counts. This involved accounting for factors such as the length of the degree descriptions and the number of skills extracted from the description, the presence of stop words, etc. By normalizing the frequency counts, I could mitigate any biases introduced by differences in the length or composition of the degree descriptions, allowing for a more accurate comparison across skills.

5.3.2 TFIDF

The first improvement for the skill scoring was to introduce a TFIDF score for each skill and recalculate the weightage of the skills based on some transformation of the TFIDF score. To transform skill text names into numerical vectors or embeddings, I utilized the skl TF-IDF vectorizer. The TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer is a commonly used technique for converting text into numerical representations. It calculates the importance of a term (in this case, a skill name) in a document (the degree texts) by considering both its frequency within the document and its rarity across the entire dataset.

The sklearn TF-IDF vectorizer works by first computing the term frequency (TF), which measures how frequently a term appears in a document relative to the total number of terms in the document. It then computes the inverse document frequency (IDF), which quantifies the rarity of a term by dividing the total number of documents by the number of documents containing that term. Finally, the TF-IDF vectorizer combines the TF and IDF scores to generate a vector representation for each skill text. Therefore TFIDF ranks skills in the context of the entire corpus of "documents" - degree descriptions, as well as within a single "document".

As an example, let's consider the skill text "3d models." Using the sklearn TF-IDF vectorizer, this skill text would be transformed into a numerical vector representation. The vector would contain values that indicate the importance of each term within the skill text, taking into account both the term frequency and the inverse document frequency. This numerical vector representation captures the essence of the skill text and can be used for further analysis and comparison with other skill vectors within the dataset. This vector is a score between 0 to 1 for each token/ word in the input.

A small example from the output of the TFIDF code is shown in 5.9. It is a very sparse matrix. Each column is a skill, and each row is a degree cluster. To make sure the multi-word skills remained intact, and the scores unbiased, I concatenated all the words of a skill with an underscore. I also converted the text to lowercase in the

3d_computer_graphics_software	3d_imaging	3d_models	3d_printing	3d_scanning	4g__telecommunication_	5s_method	ab_initio	...	world_wide_web	wound_healing
0.0	0.0	0.055486	0.0	0.0	0.0	0.0	0.000000	...	0.000000	0.000000
0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.093356	0.000000
0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.000000	0.000000
0.0	0.0	0.082421	0.0	0.0	0.0	0.0	0.000000	...	0.000000	0.000000
0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.000000	0.000000
...	...	...	...	...	...	...	...	...	...	...
0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.000000	0.000000
0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.094375	...	0.000000	0.108721
0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.000000	0.000000
0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.000000	0.000000

Figure 5.9: TFIDF output

preprocessing section. To incorporate the TFIDF scores into the final weightage system of the skill ranking, I chose to make linear transformations to the frequency weightages for each skill depending on its TFIDF score. If the score of a skill was less than 0.1, I Halved the existing "weight" and if it was more than 0.2, I doubled the "weight".

5.4 TextRank for Skill Ranking

This section explores the integration of TextRank, a graph-based ranking algorithm, into the NLP pipeline for skill ranking. It discusses the additional steps taken to refine the results, including thresholding and post-processing techniques.TextRank is a powerful algorithm that applies the concept of PageRank to identify important nodes (skills, in this case) within a graph (text document). To facilitate TextRank analysis, the skills are transformed into sentences by appending a full stop after each skill. This enables TextRank to rank each sentence based on its importance, effectively ranking the skills themselves. this "rank" is shown by giving each "sentence" a score between 0 and 1. the higher the score, the more important that "sentence" is. Therefore TextRank scores only within the context of a single "document", not across the entire corpus.

To obtain the final results and incorporate the TextRank scores into the skill ranking system, additional thresholding and post-processing steps are employed. A linear transformation is implemented. If a skill's TextRank score is below 0.05, the corresponding weight for that skill is halved. Conversely, if a skill's score exceeds 0.1, the weight is doubled. This adjustment allows skills with higher TextRank scores to carry more weight in the final skill ranking, thereby emphasizing their importance.

5.5 LLM Prompting

For the purposes of this thesis, I used OPenAI's text-davinci-003 model dav, which falls under the GPT 3.5 series. The GPT-3.5 series models were trained on a blend of text and code data. GPT-3.5-turbo is an advanced language model with enhanced performance and capabilities. It follows a transformer-based architecture, is pre-trained on diverse data, and has a large-scale model size with 175 billion parameters. GPT-3.5-turbo excels in contextual understanding, generating human-like text, and has some limited support for multimodal inputs. It can be fine-tuned for specific tasks and is widely used in natural language processing applications. GPT-3.5 is an improvement on the GPT-3 model. GPT-3 is an autoregressive language model and the third generation

in the GPT-n series. It has gained attention for its remarkable language generation abilities, with the largest version, GPT-3 175B, having 175 billion parameters, 96 attention layers, and a 3.2 million batch size.

One key aspect that influences the behaviour of these models is the technique of prompting. Prompting refers to the use of specific instructions or input patterns provided to the model to guide its output generation process. Prompting involves providing explicit instructions or cues to guide the language model's response generation. These instructions can take various forms, such as partial sentences, keywords, explicit rules, or template-based structures. Template-based prompting utilizes predefined templates or structures as prompts. These templates can be filled in with dynamic or user-specific information to generate personalized responses. For instance, a template-based prompt might include placeholders for user names, locations, or other variables, allowing the model to produce tailored output. The different prompt structures had a large impact on the generated responses. here I tried to recreate Prop paper's Alivanistos (2022) results. Two specific prompts are considered: Triple-based Prompts and Natural Language Prompts. In the following subsection, we delve into a detailed analysis of how variations in prompt structures, specifically the order of options and the wording of natural language prompts, affect the model's interpretation and generate varying degrees of ambiguity.

5.5.1 Natural Language Prompts

The formulation of natural language prompts plays a critical role in shaping the model's understanding and subsequent responses. Subtle variations in wording, such as using "generally teaches" versus "typically teaches," or incorporating synonyms like "on an average" or "usually," resulted in distinct outputs. These differences were particularly evident when addressing skills or concepts that possess inherent ambiguity, even when assessed by human annotators.

The observations above highlight the significance of carefully crafting prompts to ensure the desired specificity and accuracy in the model's responses. The selection of specific words and phrases significantly influences the model's interpretation and ability to provide accurate outputs. Conducting experiments with different prompt structures and phrasings can aid in refining the model's responses, aligning them with the desired level of clarity and reliability.

5.5.2 Triple-Based Prompts

Triple-based prompts offer an alternative approach to address the challenges associated with natural language prompts. These prompts solely contain the relevant terms necessary for predicting object entities based on subject entities and relations. By eliminating extraneous information and focusing on key elements, triple-based prompts provide a more concise and precise framework for generating predictions. Consequently, this approach mitigates the impact of irrelevant words or combinations, which often hinder the precision of model predictions when using natural language prompts.

The benefits of triple-based prompts lie in their ability to reduce the cognitive load on the model and limit the potential for misinterpretation caused by unnecessary wording. The simplified structure facilitates more accurate predictions by enabling the model to focus solely on the essential information required for generating responses.

5.5.3 Importance of Yes/No/Maybe Options

In addition to the order of options and the wording of prompts, the inclusion of yes/no/maybe options proved to be instrumental in improving the model's response quality. Initially, when only questions were asked without providing specific answer options, the model tended to generate ambiguous or lengthy responses. However, upon introducing the yes/no/maybe choices, the model's outputs became more focused and definitive.

By explicitly instructing the model to select from these options, I provided clearer guidance and reduced the likelihood of ambiguous or verbose answers. The predefined answer choices enabled the model to align its responses with the desired format and precision. Instead of generating open-ended or uncertain answers, the model could provide concise and decisive outcomes based on the available options. This approach leverages the model's capability to make binary choices and provides a more structured framework for interaction. By narrowing down the range of possible responses to yes, no, or maybe, the model gains a clearer understanding of the expected output and can produce more concrete and accurate results.

5.5.4 Order of Options

By manipulating the order of options within brackets, namely [yes, no, maybe], I observed intriguing observations regarding the model's responses. Placing "maybe" as the first option tended to produce more uncertain or inconclusive results, suggesting a higher level of ambiguity in the model's output. This finding implies that the model is inclined to lean towards an uncertain response when "maybe" is presented as the initial choice. I also explored the impact of replacing "maybe" with "sometimes" as an option. This alteration introduced an additional layer of variation in the model's output.

5.5.5 Final prompt template chosen

- Triple-based Prompts: For the Triple-based Prompts, the prompt in question is, "*degree name* degree teaches *skill name*: yes or no?"
- Natural Language Prompts: The Natural Language Prompt used is, "*degree name* degree programs typically teaches *skill name* [yes, no, maybe]?"

Chapter 6

Results

Chapter 6 provides a comparative evaluation of the baseline and further enhancements and the analysis of the obtained results.

6.1 Evaluation Methods

In this study, I chose to use precision at various top-K values (7, 10, 20, 30, and 40) as the primary evaluation method. This choice was motivated by several factors. The approach of evaluating precision at different K values aligns with established practices in similar domains. By examining precision at multiple K values, we gain a comprehensive understanding of the model's performance across different levels of skill prediction.

The dataset was annotated with binary labels indicating the presence or absence of skills for each education instance. This binary nature of the annotations allowed me to treat the skill prediction task as a classification problem, where I aimed to classify whether a particular skill exists or not for a given education instance. Consequently, precision, which measures the proportion of correctly predicted positive instances (true positives) out of all predicted positive instances (true positives and false positives), was a suitable metric for the evaluation.

The choice of precision over recall was influenced by the priorities of the company for which this model was developed. The company placed more emphasis on the existence of relevant skills rather than their ranking or order. Therefore, precision, which focuses on the correctness of positive predictions, was considered more important in this context. By utilizing precision at various top-K values, we account for the uncertainty associated with predicting the exact order or rank of skills. This approach acknowledges that the primary objective is to identify the presence of relevant skills within the top predictions, rather than their specific order. Top-K accuracy provides a more relaxed evaluation metric that captures the model's ability to identify the correct skills within a given range of predictions.

6.1.1 Top-K precision

Top-K precision is a quantitative metric used for evaluating the performance of recommendation and retrieval systems. It centres on the accuracy of a model's predictions within a specified range of top results. The number of relevant items among these top-K recommendations is tallied, and the precision score is computed by dividing this count by K. The resulting precision value offers insight into the model's efficacy in ranking

and retrieving relevant items in its top suggestions. Higher precision scores indicate more accurate recommendations. The procedure is iterated for different K values, allowing a comprehensive assessment of the model's performance across a spectrum of recommendation contexts.

6.1.2 Precision

Precision is a metric used in information retrieval and machine learning to evaluate the accuracy of a model or system in retrieving relevant results. It measures the proportion of retrieved instances that are actually relevant to the query or task at hand. Precision focuses on the quality of the retrieved results rather than the overall completeness.

Mathematically, precision is calculated as the number of true positive instances divided by the sum of true positive and false positive instances:

$$Precision = \frac{True\ Positive(TP)}{True\ Positive(TP) + False\ Positive(FP)}$$

Figure 6.1: Precision

True positives (TP) are the instances that were correctly identified as relevant, while false positives (FP) are the instances that were incorrectly identified as relevant.

A high precision value implies that a system or model has a low rate of false positives, showing that the majority of the retrieved instances are relevant. On the other hand, a low precision value suggests that there are many false positives, and the retrieved results may contain a significant number of irrelevant instances.

Precision at top	Baseline (Co-occurr + Frequency)	Base + TFIDF	Base + TextRank (TR)	Base + TR + Prompt 1	Base + TR + Prompt 1+2	Test Results: Base + TR +Prompt 1+2
7 skills	0.607	0.593	0.629	0.842	**0.857**	**0.840**
10 skills	0.605	0.635	0.64	0.820	0.815	0.818
20 skills	0.552	0.580	0.585	0.672	0.712	0.706
30 skills	0.536	0.550	0.545	0.600	0.628	0.609
40 skills	0.500	0.519	0.515	0.515	0.537	0.522

Table 6.1: Dev and Test Set Results

6.1.3 Baseline

In order to establish a benchmark for comparison, a frequency baseline approach was employed. This approach utilized the occurrence of skills in the degree descriptions and assigned weights to the skills based on their frequency within the degrees. The higher the frequency of skills in the degree descriptions, the more important it was considered in the ranking. The baseline results revealed several significant findings and observations. The precision values obtained from the frequency-based ranking approach showed a decline as the number of skills included in the ranking increased.

Specifically, the precision values at the top 7 and 10 skills were relatively higher (0.607 and 0.605, respectively) compared to the precision values at the top 20, 30, and 40 skills (0.552, 0.536, and 0.500, respectively). This suggests that the accuracy of the top-k precision measuring approach decreases as more skills are included in the ranking. While larger K values encompass more diversity in recommendations, they also make it more challenging to maintain a consistently high level of accuracy, potentially resulting in a reduction of precision. The frequency-based approach may overestimate the importance of certain skills, especially as the ranking expands. Therefore, alternative techniques that consider additional factors beyond frequency should be explored. A comparative analysis with other ranking methods is necessary to assess the relative effectiveness of the frequency-based approach. These findings serve as a foundation for further investigation and refinement of the methodology to enhance the precision and accuracy of skill ranking within degree descriptions.

6.1.4 Statistical Enhancements

To improve the baseline approach, two statistical methods, TDIDF score and Textrank score, were incorporated, with a linear modification of the baseline weights. The modified scores, namely Baseline+TDIDF and Baseline+TextRank scores, were calculated and evaluated. The precision values obtained from the enhanced approach using TDIDF scores showed improvements across different skill rankings, with an increase of 1% on average. Similarly, the precision values from the enhanced approach using Textrank scores demonstrated further enhancements. The precision values increased by an average of 2% compared to the baseline approach. The Textrank method considers the co-occurrence and contextual relationships between skills to determine their significance. The enhanced precision values obtained with baseline+textrank indicate that Textrank successfully captures the interconnectedness of skills and their relevance within the degree descriptions.

The TDIDF method primarily considers the frequency of skills in degree descriptions, while the Textrank method incorporates semantic relationships and contextual relevance. Textrank's ability to capture the broader context and relevance of skills may explain its superior performance.

6.1.5 LLM Prompting Results

The experiment explored two prompting techniques, natural language prompts and triple-based prompts. Natural language prompts provided flexibility in phrasing and allowed for more contextual instructions, while triple-based prompts focused on key subject entities and relations. Both techniques resulted in comparable precision values, demonstrating the effectiveness of both approaches. On closer analysis, they each had their strengths and therefore shows that a combination of the two would work together even better than any one individually. This is the final result that performed best.

Subtle variations in wording within the natural language prompts had noticeable effects on the model's interpretation and responses. Different phrasings led to distinct outputs, especially when dealing with ambiguous skills or concepts 5.5 . This finding underscores the importance of carefully crafting prompts to ensure specific and accurate responses. Including yes/no/maybe options significantly enhanced the quality of the model's responses. By providing predefined answer choices, the guidance provided to the model became clearer, resulting in more focused and definitive outputs. The use of

structured answer options helped reduce ambiguity and encouraged more concise and accurate responses from the model.

The LLM's exposure to diverse training data, including text and code, allows it to capture a wide range of domain-specific information. This knowledge base enhances its understanding of degree programs, their associated skills, and the specific requirements of different fields of study. As a result, the LLM is well-equipped to make informed predictions and generate accurate skill rankings within degree descriptions.

6.1.6 Test Set Results

Across all skill rankings, the precision values obtained from the test set are generally lower than those from the dev set. This indicates a slight drop in performance when the models are evaluated on unseen data (the test set). Despite the drop in precision, the performance of the models on the test set remains relatively consistent across different skill rankings. This suggests that the models' ranking capabilities are relatively robust and not heavily influenced by the specific number of skills considered in the ranking.

6.2 Error Analysis

To gain a deeper understanding of the errors made by the models in skill ranking within degree descriptions, an error analysis was conducted. The analysis focused on examining the types of errors made and identifying potential patterns or underlying causes.

degree_topic	perc7	perc10	perc20	perc30	perc40
Archaeology Research	0.71	0.7	0.45	0.47	0.4
Architecture	0.71	0.8	0.7	0.6	0.5
Artificial Intelligence	0.86	0.9	0.8	0.8	0.7
Arts Literature Media	0.29	0.5	0.4	0.4	0.3
Bio Pharmaceutical Sciences	1	1	0.8	0.7	0.65
Biology	0.86	0.9	0.75	0.6	0.52
Business Studies	1	0.9	0.9	0.77	0.7
Cyber Security	1	0.8	0.8	0.83	0.78
Ecology	1	1	0.6	0.53	0.42
Economics	0.86	0.9	0.65	0.63	0.48
Education Child Studies	0.86	0.9	0.85	0.7	0.62
English Language Culture	0.86	0.6	0.5	0.4	0.38
International Studies	0.86	0.8	0.5	0.43	0.35
Law	1	1	1	0.87	0.65
Marketing	1	0.7	0.5	0.37	0.28
Mathematics	1	1	0.85	0.7	0.55
Medicine	1	1	0.95	0.93	0.8
Music Communication Technology	0.57	0.5	0.3	0.3	0.28
Physics program	1	0.9	0.75	0.63	0.6
Psychology	0.71	0.7	0.65	0.53	0.45

Figure 6.2: Error Analysis: Precision at Top k Skills

Some errors were attributed to the inherent ambiguity present in the degree descrip-

tions. Certain descriptions may have been vague or open to interpretation, making it challenging for the models to accurately identify and rank the relevant skills. Ambiguity in the descriptions could stem from the use of broad terms, generalized statements, or lack of explicit skill references, thereby impacting the precision of the predictions. Another category of errors involved the models failing to identify certain skills that were indeed present in the degree descriptions. Missing skills could also be attributed to the limited training data or biases within the dataset, resulting in overlooked or underrepresented skills. Degree Topic "Economics" achieved high precision (1.0) 6.2 for the top 7 skills but experienced a decline in precision as the number of skills considered increased. This suggests that the models might have failed to identify and rank some relevant skills beyond the initial subset. The models might have limitations in recognizing subtle variations or synonyms of skills specific to economics, leading to missing skill errors as the ranking expands to include a larger number of skills.

The choice of the skill ranking threshold also influenced the precision values. For example, the degree topic "Physics Program" achieved perfect precision when considering the top 7 and top 10 skills but experienced a decline in precision as the number of skills considered increased. This suggests that the models might struggle to rank a larger number of skills accurately within the degree descriptions. Some degree topics exhibited unusually low or high precision or recall values compared to others. For instance, the degree topic "Marketing" achieved a high precision of 1 for the top 7 skills 6.2 but had lower precision for the subsequent skill rankings. This shows the potential challenges in accurately ranking skills beyond the initial subset. STEM Courses also seem to do better than Humanities Courses.

The degree Topic "Arts Literature Media" exhibited lower precision values 6.3 across all skill rankings. This demonstrates the challenges in accurately ranking skills within degree descriptions for this topic, possibly due to the inherent ambiguity in descriptions related to arts, literature, and media. The degree descriptions in this topic may contain generalized statements, broad terms, or open-ended references to skills, making it difficult for the models to precisely identify and rank the relevant skills.

Degree Topic "Psychology" achieved moderate precision values across various skill rankings. This suggests that the prompts used to guide the models might not have been fully effective in capturing the specific requirements or nuances of psychology-related degree descriptions. The prompts may have lacked the necessary specificity or context to accurately guide the models in identifying and ranking the most relevant skills within psychology degree descriptions.

degree_topic	perc7_pos	perc10_pos	perc20_pos	perc30_pos	perc40_pos
Archaeology Research	0.28	0.39	0.5	0.78	0.89
Architecture	0.23	0.36	0.64	0.82	0.91
Artificial Intelligence	0.19	0.28	0.5	0.75	0.88
Arts Literature Media	0.17	0.42	0.67	1	1
Bio Pharmaceutical Sciences	0.23	0.32	0.52	0.68	0.84
Biology	0.26	0.39	0.65	0.78	0.91
Business Studies	0.23	0.3	0.6	0.77	0.93
Cyber Security	0.2	0.23	0.46	0.71	0.89
Ecology	0.33	0.48	0.57	0.76	0.81
Economics	0.32	0.47	0.68	1	1
Education Child Studies	0.19	0.28	0.53	0.66	0.78
English Language Culture	0.35	0.35	0.59	0.71	0.88
International Studies	0.35	0.47	0.59	0.76	0.82
Law	0.27	0.38	0.77	1	1
Marketing	0.58	0.58	0.83	0.92	0.92
Mathematics	0.32	0.45	0.77	0.95	1
Medicine	0.18	0.26	0.5	0.74	0.84
Music Communication Technology	0.36	0.45	0.55	0.82	1
Physics program	0.26	0.33	0.56	0.7	0.89
Psychology	0.28	0.39	0.72	0.89	1

Figure 6.3: Error Analysis: Percentage Precision WRT Total Annotated Positive Examples

degree_topic	top7	top10	top20	top30	top40	tot_pos	tot
Archaeology Research	5	7	9	14	16	18	50
Architecture	5	8	14	18	20	22	50
Artificial Intelligence	6	9	16	24	28	32	50
Arts Literature Media	2	5	8	12	12	12	32
Bio Pharmaceutical Sciences	7	10	16	21	26	31	50
Biology	6	9	15	18	21	23	50
Business Studies	7	9	18	23	28	30	50
Cyber Security	7	8	16	25	31	35	50
Ecology	7	10	12	16	17	21	50
Economics	6	9	13	19	19	19	30
Education Child Studies	6	9	17	21	25	32	50
English Language Culture	6	6	10	12	15	17	50
International Studies	6	8	10	13	14	17	50
Law	7	10	20	26	26	26	39
Marketing	7	7	10	11	11	12	50
Mathematics	7	10	17	21	22	22	50
Medicine	7	10	19	28	32	38	50
Music Communication Technology	4	5	6	9	11	11	50
Physics program	7	9	15	19	24	27	50
Psychology	5	7	13	16	18	18	39

Figure 6.4: Error Analysis: Raw Numbers

Chapter 7

Discussion and Future Directions

7.1 Summary of the research

This thesis project aimed to explore how university curricula and course descriptions can be leveraged to improve the quality of degree-skill relationships in knowledge graphs. The research question focused on understanding the methods that can facilitate the extraction and enrichment of this information.

To address this research question, a pipeline consisting of three main steps was developed: data collection and annotation, graph building/enrichment, and evaluation. The data collection process involved scraping text from university course catalogues to gather information about degree programs and their associated courses. Semi-automated annotation techniques were employed to label the data and establish relationships between degrees and skills. The graph building/enrichment step utilized similarity measures, such as word embeddings and TF-IDF clustering, to associate skills with degree programs. The enriched graph was evaluated using subtree similarity scores and compared against hand-created ground truth data.

The results of the research demonstrated improvements in the precision of skill ranking within degree descriptions compared to a baseline frequency-based approach. The incorporation of statistical methods, such as TF-IDF and TextRank, enhanced the accuracy of the ranking process. Furthermore, the use of language model prompting techniques, including natural language prompts and triple-based prompts, contributed to refining the models' responses and generating more focused outputs.

7.2 Answer the research question

The research question, "How can university curricula and course descriptions be leveraged to improve the quality of degree-skill relationships in knowledge graphs, and what methods can facilitate the extraction and enrichment of this information?" has been addressed through the implementation of the proposed pipeline and the evaluation of its results.

The research findings demonstrate that by leveraging university curricula and course descriptions, it is possible to enrich knowledge graphs with degree-skill relationships. The use of statistical techniques, such as TF-IDF and TextRank, and the incorporation of language model prompting methods contribute to enhancing the accuracy and precision of skill ranking within degree descriptions.

Subquestions 1 and 2: Evaluating the effectiveness of the enriched graph in matching skills to degree programs involves various approaches. Precision can be calculated to measure the accuracy and completeness of the matches. Top-K accuracy assesses the percentage of correct matches within the top predicted skills for each degree program. Mean Average Precision (MAP) provides an overall measure of performance by considering the average precision across all skills and degree programs. Human evaluation can also offer qualitative insights by comparing matches with ground truth data or using subjective judgments. The performance of the final approach far exceeded the results of the baseline approach, as seen in 6.1.

Subquestion 3: Matching fresh graduates' skills with degree programs based on university curricula and course descriptions presents specific limitations and challenges. Ambiguity in course descriptions, stemming from vague or generalized language, can hinder accurate skill identification. The limited coverage of skills in curricula, with some skills being implicitly taught or not explicitly listed, adds complexity. Variations across universities, such as terminology and skill emphasis, require context and domain-specific knowledge consideration. Limited availability and inconsistent quality of data also pose challenges in achieving accurate matches.

7.3 Challenges faced

One significant challenge faced during the development of this system was the availability and quality of labelled data. The limited availability of annotation resources and tools posed a challenge, as the creation of high-quality annotations required access to suitable platforms and expertise in the domain. The issue of annotated data bias also emerged as a difficulty, as the availability of properly labelled data directly affected the system's performance. Ensuring a balanced and unbiased representation of the annotated data is crucial to avoid potential biases in the recommendation process. Even data collection posed a significant obstacle, as obtaining a sufficient quantity of relevant data from diverse sources proved to be time-consuming and resource-intensive. The implementation of a skill service proved to be challenging, requiring careful consideration of skills and their mapping to specific programs and courses. And finally, the access to GPT-3.5 and its associated resources presented limitations and constraints on the system's development. The availability of more comprehensive resources and access to advanced models would be beneficial for further improving the recommendation system.

In spite of these challenges, this thesis has made significant progress. The exploration of future work and identification of the difficulties faced provide valuable insights for researchers and practitioners in the field, guiding further advancements and overcoming obstacles in the development of personalized and accurate systems for HRM.

7.4 Future work

Throughout the project, several future directions and challenges were identified, which can serve as valuable insights for further research and development in this field.

One of the potential future works is to implement the Term Frequency-Inverse Document Frequency (TF-IDF) approach within a program and its courses, rather than considering it solely across different programs. This would allow for a more fine-

grained analysis and recommendation process, considering the specific content and characteristics of individual courses within a program.

Another aspect that could contribute to enhancing the recommendation system is the utilization of a larger and more powerful model. While the current model used in this thesis has provided satisfactory results, a bigger and better model could potentially offer even more accurate and diverse recommendations. Augmenting the existing data with additional relevant data sources would also contribute to improving the system's performance. Experimenting with different prompts, question formats, and variations in input, it may be possible to refine the recommendation system and extract more nuanced and specific information from users, resulting in more tailored and personalized recommendations.

There could be other approaches that can be taken in the annotation process. To move beyond binary responses of "yes" or "no," incorporating a scoring model, such as sentiment analysis or ranking, could provide a more detailed and informative feedback mechanism. This would enable users to express their preferences and priorities in a more nuanced manner, leading to better-tailored recommendations that align with their specific needs and goals. Implementing feedback loops and active learning strategies can also enable continuous refinement of the knowledge graph and improve its effectiveness over time.

In conclusion, this thesis project has explored the leveraging of university curricula and course descriptions to improve the quality of degree-skill relationships in knowledge graphs. Through the development of an enriched graph and the evaluation of its effectiveness, significant contributions have been made to the fields of HRM and NLP. However, there are still areas that can be further investigated in future work. By continuing to advance and refine this research, we can foster innovation in the field and contribute to the successful transition of graduates into the workforce.

Appendix A

Appendix Title

GitHub Link to the code: Git

A.1 Education Skill Relation Annotation Instructions and Guidelines

In this task you are given a set of Degree topics and, for each degree, a set of skills that we believe are typical/important. Your task is to determine whether this is indeed the case. In particular, for a given degree you are provided with a spreadsheet with the following information:

- The degree's name and level of education. There are a few columns at the end ('cluster_ignore', 'weight_y_ignore', 'source_ignore') that are to be ignored for the process of annotation, but need to be kept in the sheet.
- A Google Search can help you to understand better what the degree is about if required.
- A set of related skills. For each skill you can see the following fields: The skill's name , An annotation field (is_degree_related_to_skill) that is empty.

Your task is to fill/edit the annotation field. This should be done as follows:

-
- If the skill is related to the degree use the value 1
- If the skill is not related to the degree use the value 0

To decide whether a skill is related to a degree, think that we are looking for skills that:

- If the skill is not related to the degree use the value 0 Are core parts of the degree's definition E.g., A Law degree teaches "Law", "European Union Laws", "Legal Knowledge", "Criminal Codes", etc
- Most degrees in this field(are expected to) teach without being obligatory. E.g., "Python" for "Computer Science", "Administrative Operations" for "International Business", etc.

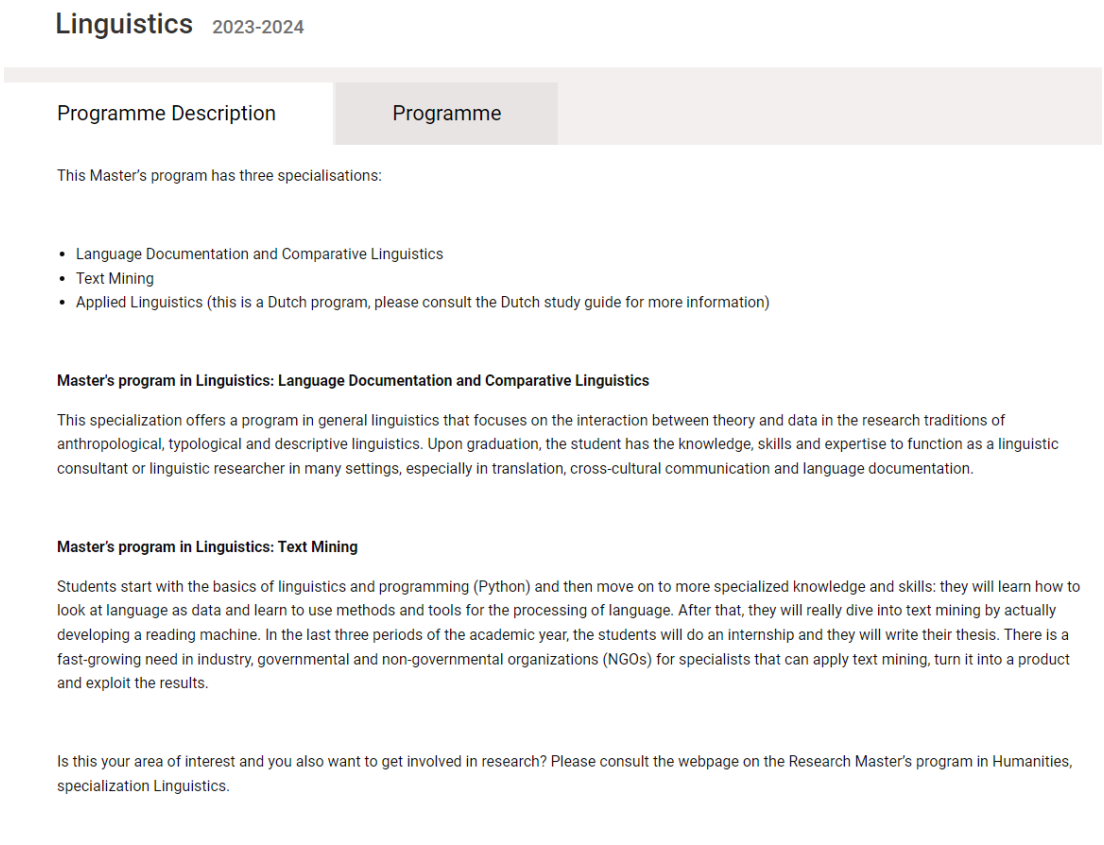
- Are not too abstract/generic/ambiguous for the specificity level of the profession E.g., “Engineering” for “Civil engineering” is a bad skill because there are many types of engineering. Similarly “Biology” for “Neuroscience” E.g. “Consulting” for “Marketing” because many degrees can teach how to consult, and consulting can be done in many other domains.
- Are not too specific for the specificity level of the profession E.g., “IBM System I” for “Finance” is a bad skill because there are many tools that can be taught in finance that can replace IBM System I.
- Are indeed skills and not some other requirement or perk E.g.: “Flexible Working”, “Heart Rate” is not a skill
- Are not specific to a particular country (e.g. only “US Civil Law codes” for “Law”)

The “0” annotation should especially be used for cases where:

- The skill is not really a skill E.g.: “Feedback Management”, ”Justice”
- The skill that is very low level or not very specific to the Degree: E.g., “Knowledge of Engineering” for “Materials science engineering”, “Java” for “Truck Driver”, etc
- The relation is embarrassingly wrong: E.g., “Cooking Skills” or “Banking Services” for “Materials science engineering”, “Knowledge of Laws” or “Medical Emergencies” for “Media Studies Cultural Analysis Literature Theory”, “Metalworking” or “Drones” for “Law” etc
- The skill is too abstract/ambiguous, independent of the degree E.g.: “Writing reports”, ”Social media”, ”Presentations”, ”Research Skills”

Focus and time spent:

- We expect most suggested skills for a degree to be somewhat related, so the focus should be on identifying the completely unrelated and odd skills that might have been wrongly suggested.
- Use the provided Google search only for degrees and skills you are not sure what they are about.
- We don’t try to assess and judge the relevant relatedness of a skill to a degree with respect to that of the other skills; we merely want to know if a skill can be considered adequately related to the degree, based on the above criteria.
- For a given degree we might have pairs of related skills where one is more specialized than the other (e.g. “Machine Learning” and “Supervised Machine Learning”, “Programming Languages” and “Java”, “Sales” and “Sales Strategy”). In such a case both skills should be selected, as long as of course, they satisfy the individual criteria mentioned above.
- We expect the average time to annotate a single-degree concept to be between 2 and 6 minutes; if it’s significantly longer than that for several degrees then let us know so as to discuss and clarify the guidelines and criteria.



Linguistics 2023-2024

Programme Description | Programme

This Master's program has three specialisations:

- Language Documentation and Comparative Linguistics
- Text Mining
- Applied Linguistics (this is a Dutch program, please consult the Dutch study guide for more information)

Master's program in Linguistics: Language Documentation and Comparative Linguistics

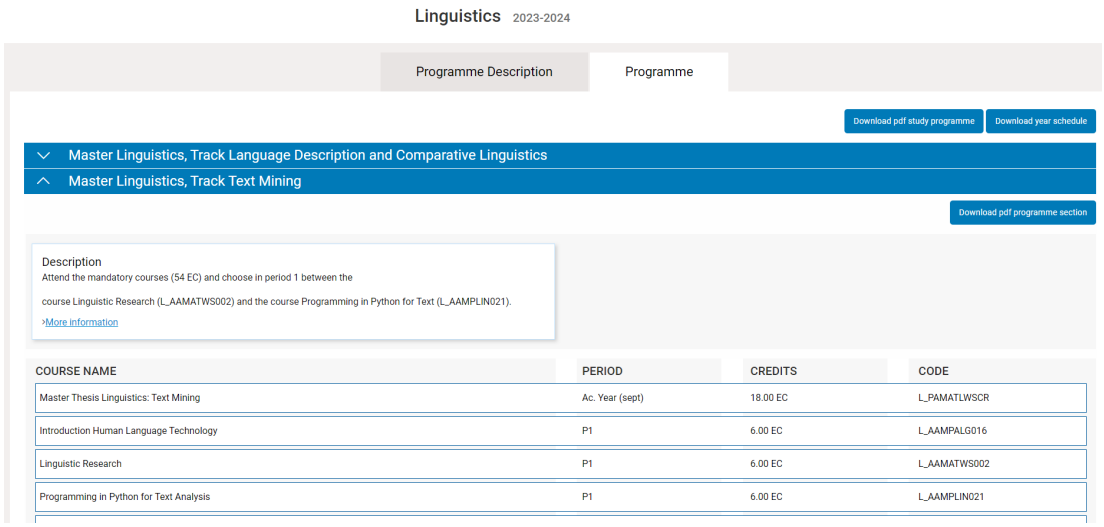
This specialization offers a program in general linguistics that focuses on the interaction between theory and data in the research traditions of anthropological, typological and descriptive linguistics. Upon graduation, the student has the knowledge, skills and expertise to function as a linguistic consultant or linguistic researcher in many settings, especially in translation, cross-cultural communication and language documentation.

Master's program in Linguistics: Text Mining

Students start with the basics of linguistics and programming (Python) and then move on to more specialized knowledge and skills: they will learn how to look at language as data and learn to use methods and tools for the processing of language. After that, they will really dive into text mining by actually developing a reading machine. In the last three periods of the academic year, the students will do an internship and they will write their thesis. There is a fast-growing need in industry, governmental and non-governmental organizations (NGOs) for specialists that can apply text mining, turn it into a product and exploit the results.

Is this your area of interest and you also want to get involved in research? Please consult the webpage on the Research Master's program in Humanities, specialization Linguistics.

Figure A.1: Broad Program Description



Linguistics 2023-2024

Programme Description | Programme

Download pdf study programme | Download year schedule

Master Linguistics, Track Language Description and Comparative Linguistics

Master Linguistics, Track Text Mining

Download pdf programme section

Description

Attend the mandatory courses (54 EC) and choose in period 1 between the

course Linguistic Research (L_AAMATWS002) and the course Programming in Python for Text (L_AAMPLIN021).

>More information

COURSE NAME	PERIOD	CREDITS	CODE
Master Thesis Linguistics: Text Mining	Ac. Year (sept)	18.00 EC	L_PAMATLWSCR
Introduction Human Language Technology	P1	6.00 EC	L_AAMPALG016
Linguistic Research	P1	6.00 EC	L_AAMATWS002
Programming in Python for Text Analysis	P1	6.00 EC	L_AAMPLIN021

Figure A.2: Course List of Program

Programming in Python for Text Analysis 2023-2024

Course Objective

Goals of this course:
• Get to know the basics of the Python programming language
• Make a start with becoming an independent programmer, who is able to find solutions to new problems

Skills you will acquire during this course:
- Learn how to develop Python code using Jupyter notebooks as well as Python modules (.py files)
- Learn how to create readable code that can be understood by others
- Learn how to debug your code
- Learn how to write pseudo code
- Learn how to make your own code project
- Learn how to deal with unstructured textual data
- Learn how to perform linguistic processing with established NLP pipelines

Course Content

During this course, you will learn how to analyze text data using the Python programming language. No programming knowledge is required; we believe that anyone can learn how to program.

You will learn how to extract information from text corpora; deal with different file types (plain text, CSV, JSON, xml). We will focus on readability and understandability of your code so that you will be able to share it with others, and reuse your code in the future.

Additional Information Teaching Methods

The course is organized in blocks. Blocks typically follow this routine:
Lecture 1: introduction of concepts in the form of an (interactive)

Figure A.3: Course Description Text

degree_topic	top7	top10	top20	top30	top40	tot_pos	tot	perc7	perc10	perc20	perc30	perc40	perc7_pos	perc10_pos	perc20_pos	perc30_pos	perc40_pos	perc7_tot	perc10_tot	perc20_tot	perc30_tot	perc40_tot
Archaeology Research	5	7	9	14	16	18	50	0.71	0.7	0.45	0.47	0.4	0.28	0.39	0.5	0.78	0.89	0.1	0.14	0.18	0.28	0.32
Architecture	5	8	14	18	20	22	50	0.71	0.8	0.7	0.6	0.5	0.23	0.36	0.64	0.82	0.91	0.1	0.16	0.28	0.36	0.4
Artificial Intelligence	6	9	16	24	28	32	50	0.86	0.9	0.8	0.8	0.7	0.19	0.28	0.5	0.75	0.88	0.12	0.18	0.32	0.48	0.56
Arts Literature Media	2	5	8	12	12	12	32	0.29	0.5	0.4	0.4	0.3	0.17	0.42	0.67	1	1	0.06	0.16	0.25	0.38	0.38
Bio Pharmaceutical Sciences	7	10	16	21	26	31	50	1	1	0.8	0.7	0.65	0.23	0.32	0.52	0.68	0.84	0.14	0.2	0.32	0.42	0.52
Biology	6	9	15	18	21	23	50	0.86	0.9	0.75	0.6	0.52	0.26	0.39	0.65	0.78	0.91	0.12	0.18	0.3	0.36	0.42
Business Studies	7	9	18	23	28	30	50	1	0.9	0.9	0.77	0.7	0.23	0.3	0.6	0.77	0.93	0.14	0.18	0.36	0.46	0.56
Cyber Security	7	8	16	25	31	35	50	1	0.8	0.8	0.83	0.78	0.2	0.23	0.46	0.71	0.89	0.14	0.16	0.32	0.5	0.62
Ecology	7	10	12	16	17	21	50	1	1	0.6	0.53	0.42	0.33	0.48	0.57	0.76	0.81	0.14	0.2	0.24	0.32	0.34
Economics	6	9	13	19	19	19	30	0.86	0.9	0.65	0.63	0.48	0.32	0.47	0.68	1	1	0.2	0.3	0.43	0.63	0.63
Education Child Studies	6	9	17	21	25	32	50	0.86	0.9	0.85	0.7	0.62	0.19	0.28	0.53	0.66	0.78	0.12	0.18	0.34	0.42	0.5
English Language Culture	6	6	10	12	15	17	50	0.86	0.6	0.5	0.4	0.38	0.35	0.35	0.59	0.71	0.88	0.12	0.12	0.2	0.24	0.3
International Studies	6	8	10	13	14	17	50	0.86	0.8	0.5	0.43	0.35	0.35	0.47	0.59	0.76	0.82	0.12	0.16	0.2	0.26	0.28
Law	7	10	20	26	26	26	39	1	1	1	0.87	0.65	0.27	0.38	0.77	1	1	0.18	0.26	0.51	0.67	0.67
Marketing	7	7	10	11	11	12	50	1	0.7	0.5	0.37	0.28	0.58	0.58	0.83	0.92	0.92	0.14	0.14	0.2	0.22	0.22
Mathematics	7	10	17	21	22	22	50	1	1	0.85	0.7	0.55	0.32	0.45	0.77	0.95	1	0.14	0.2	0.34	0.42	0.44
Medicine	7	10	19	28	32	38	50	1	1	0.95	0.93	0.8	0.18	0.26	0.5	0.74	0.84	0.14	0.2	0.38	0.56	0.64
Music Communication Technology	4	5	6	9	11	11	50	0.57	0.5	0.3	0.3	0.28	0.36	0.45	0.55	0.82	1	0.08	0.1	0.12	0.18	0.22
Physics program	7	9	15	19	24	27	50	1	0.9	0.75	0.63	0.6	0.26	0.33	0.56	0.7	0.89	0.14	0.18	0.3	0.38	0.48
Psychology	5	7	13	16	18	18	39	0.71	0.7	0.65	0.53	0.45	0.28	0.39	0.72	0.89	1	0.13	0.18	0.33	0.41	0.46

Figure A.4: Error Analysis

Italy
University of Pisa
France
University of Grenoble Alpes
Université Paris Cité
University of California Education Abroad Program
Switzerland
École Polytechnique Fédérale de Lausanne
Poland
The University of Warsaw
University of Wroclaw
Sweden
Umeå University
Lund University
Norway
UiT The Arctic University of Norway
University of Oslo
Norwegian University of Science and Technology

Table A.1: List of Universities

Bibliography

Formula. URL https://datatab.net/tutorial/fleiss-kappa.

Github link. URL https://github.com/cltl-students/Saloni_Singh_Master_thesis_2023/tree/main.

URL https://platform.openai.com/docs/model-index-for-researchers/models-referred-to-as-gpt-3-5.

Package. URL https://pypi.org/project/cleantext/.

URL https://platform.openai.com/docs/models.

Package. URL https://pypi.org/project/kmeans/.

Package. URL https://docs.python.org/3/library/re.html.

Package. URL https://pypi.org/project/requests/.

URL https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html.

URL https://scikit-learn.org/stable/.

URL https://spacy.io/api/doc.

URL https://pypi.org/project/pytextrank/.

Aug 2023. URL https://en.wikipedia.org/wiki/Cluster_analysis#:~:text=Cluster%20analysis%20or%20clustering%20is,in%20other%20groups%20(clusters).

Aug 2023. URL https://en.wikipedia.org/wiki/Cohen%27s_kappa.

Aug 2023. URL https://en.wikipedia.org/wiki/K-means_clustering.

Aug 2023. URL https://en.wikipedia.org/wiki/PageRank.

D. Alivanistos. Prompting as probing: Using language models for knowledge base construction. *CEUR Workshop Proceedings*, 3274, 2022. URL https://ceur-ws.org/Vol-3274/paper2.pdf.

T. B. Brown and B. Mann. Language models are few-shot learners, Jul 2020. URL https://arxiv.org/abs/2005.14165.

K. Chen. Introduction to natural language processing-tf-idf. 2021. URL https://kinder-chen.medium.com/introduction-to-natural-language-processing-tf-idf-1507e907c19.

N. Heist. Towards knowledge graph construction from entity co-occurrence. *CEUR Workshop Proceedings*, 2306, 2018. URL https://ceur-ws.org/Vol-2306/paper9.pdf.

K. C. Hyun-Jin Kim. Optimization of associative knowledge graph using tf-idf based ranking score. 2020. URL https://www.mdpi.com/2076-3417/10/13/4590.

Z. Liu. Clustering to find exemplar terms for keyphrase extraction. 2009. URL https://aclanthology.org/D09-1027.pdf.

P. T. Rada Mihalcea. Textrank: Bringing order into texts. 2004. URL https://web.eecs.umich.edu/~mihalcea/papers/mihalcea.emnlp04.pdf.

M. Rouse. Large language model (llm), Jul 2023. URL https://www.techopedia.com/definition/34948/large-language-model-llm#:~:text=A%20large%20language%20model%20(LLM)%20is%20a%20type%20of%20machine,from%20one%20language%20to%20another.

Sciforce. What is gpt-3, how does it work, and what does it actually do?, Sep 2021. URL https://medium.com/sciforce/what-is-gpt-3-how-does-it-work-and-what-does-it-actually-do-9f721d69e5c1.

Textkernel. Textkernel's usp, 2023. URL https://www.textkernel.com/how-textkernels-matching-technology-speeds-up-manpowers-placements-in-switzerland/.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, Aug 2023. URL https://arxiv.org/abs/1706.03762.

J. Ye and X. Chen. A comprehensive capability analysis of gpt-3 and gpt-3.5 series models. Mar 2023. URL https://arxiv.org/abs/2303.10420.