Developing a Natural Language Interface for Knowledge Graphs

Ruth Assefa, Sarah Mendoza
Computer Science
Lyle School of Engineering
Dallas, United States
rassefa@smu.edu
sarahmendoza@smu.edu

Luke Voinov
Electrical Computer Engineering
Lyle School of Engineering
Dallas, United States
lvoinov@smu.edu

Oyku Serap Ogut
Computer & Data Science
Lyle School of Engineering
Dallas, United States
oogut@smu.edu

Dr. Nurcan Yuruk
Computer Science
Lyle School of Engineering
Dallas, United States
nyuruk@smu.edu

***Abstract*— This paper addresses the challenges users face when querying databases. We developed an interface using OpenAI's ChatGPT-3.5 model for users to query the database in English. The database stores relations between researchers at Southern Methodist University (SMU) and groups researchers with similar researchers or research areas. The database stores data from 184 researchers across 14 departments, each containing their areas of expertise and biographies. We transformed these descriptions into word embeddings and grouped researchers by the highest cosine similarity to the queried topic. We enabled natural language queries by connecting OpenAI's ChatGPT 3.5 with the database via the Langchain Python library. The interface provided an easy and accurate way for users to interact with the database. Queries through the LLM were more robust than those made through code, indicating that ease-of-use came with increased accuracy.**

***Keywords*— knowledge graph, database, large language model, retrieval augmented generation, natural language, word embeddings, OpenAI, Neo4j, Langchain, user interfacing**

I. Introduction

Databases contain a wealth of information that is often inaccessible to would-be users who do not know the coding language required for exploring - or querying - the data. If a company intends to distribute this database, they confront the question of how their customers can use it. Even if the database fulfills its purpose, customers will shrink from the demand of learning unforgiving coding techniques. A natural solution would be to replace querying in code with querying in English. Such an approach could be implemented with algorithms but would require many preconfigured questions phrased in a certain way. Large Language Models (LLMs) provide a promising alternative due to their natural language comprehension and flexibility [1]**.** We interfaced the database with OpenAI's ChatGPT 3.5 model, allowing users to query the database in English.

Our database consists of 184 researchers across 14 departments at Southern Methodist University (SMU). This data is stored and organized as "nodes" in a Neo4j graph database. Each researcher node (if the data is not initially absent) contains a researcher's reported "research areas" and "biography." To test the LLM interface on a practical database, we addressed a sub-problem: queries should yield the most relevant data points. We used OpenAI's API to create word embeddings of each researcher's biographies and research interests. These embeddings were added to their respective nodes, and we used cosine distance to assess semantic similarity. We connected the interface to ChatGPT 3.5 through Langchain using Retrieval Augmented Generation (RAG) techniques.

To ensure validity, we imported and used the most informative data and refined the interface to retrieve and generate answers more accurately.

The LLM interface is robust in the following ways:

1) Users can query the database in English.
2) The LLM discovers high accuracy relations that would take humans a long time to find, if ever.
3) The interface groups researchers from across disciplines.
4) The interface finds associations between researchers hidden to the coded queries.

Thus, the interface is user-friendly and was able to find more varied and accurate groupings than would be found through a coded query.

II. Methods

Process Overview

We extracted information about each professor's research interests from their profiles found on SMU websites. We extracted data from 14 different webpages, each representing a different department within SMU. Half of them were from the Lyle School of Engineering and the rest were from the Dedman School of Humanities and Sciences. Information about education, research areas, biographies, and recent publications were present for most of the researchers. We used only research areas and biographies in our database since these contained all the relevant information about research interests. Incorporating education would confound whether researchers were grouped due to research similarity or due to similar educational backgrounds. Recent publications would confound the search with narrow research questions. Furthermore, the length of recent publications was highly variable between

professors, which would bias professors with lengthier statements. We parsed the relevant data from the websites with a Python library, Beautiful Soup. The information was saved as a Pandas data frame and uploaded to Excel comma-separated value (CSV) files. We extracted information from 184 researchers in total. See Appendix III for websites and specific departments.

We used these CSVs to create a relationship graph in Neo4j. The graph has the following nodes: “University”, “School”, “Department”, and “Researcher”. Each node contains properties relevant to the node type (e.g. “Researcher” nodes contain the properties ‘name’, ‘biography’, ‘research areas’, and ‘department’). Nearly every node in the database had at least one relationship with another node, mainly for organizational purposes. We added a final word embedding property to each researcher's node. The embeddings were generated through OpenAI’s API.

Creating a Neo4j Graph

Neo4j is a database that stores data as a knowledge graph. The fundamental units of a knowledge graph are the node and the relation. Neo4j graphs were established upon ideas from graph theory [2], so graph theory terminology and concepts (e.g. directed, weighted) apply. Neo4j is known for its efficient data storage, clear visualization of structures, and intuitive navigation.

Nodes

A node represents some object such as companies or people (Fig. 1). They appear as colored-in circles, where the color arbitrarily represents the type of node it is. The node of type “Department” will look and behave differently from “Researcher”. Besides having a type, nodes usually contain multiple properties.

Relations

Relations show how nodes are connected to each other. They are shown as arrows pointing from one node to another. Like nodes, they each have their own type (Fig. 1) and must have at least one property.

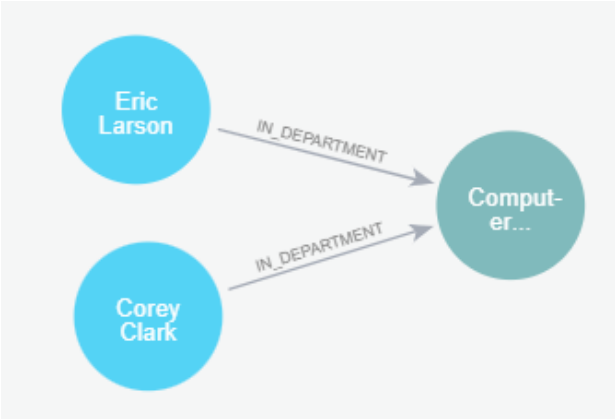

Figure 1: The nodes “Eric Larson” (E) and “Corey Clark” (C) of type “Researcher” are distinct from each other, and distinct from “Computer Science” (CS) of type “Department”. E and C contain properties such as “name” and “bio”. E and C are not related to each other, but both are related to CS because of the relation “IN_DEPARTMENT”. Note that relations are one way.

Schema of our Graph

Our graph has 5 distinct nodes: Researcher, Department, School, University, and Query. It also has 3 distinct relations: IN_DEPARTMENT, DEPARTMENT_OF, and IN_UNIVERSITY. Each of these nodes, except for the Query node, is connected via one of these relations (Fig. 2). Since all the researchers were gathered from SMU, the University node is redundant until we gather data from other universities. The most important nodes are the Researcher and Query nodes. These nodes answer questions about similarity; all the other nodes serve an organizational purpose. Relations between nodes also serve an organizational purpose.

Each Researcher node has multiple properties, the most relevant being the “name”, “bio”, “researchAreasString”, “researchEmbeddingLARGE”, and “bioEmbeddingLARGE” (see Appendix I for a list of each node’s properties). If the researcher did not report their research areas or biographies, their node contains neither those properties nor the embeddings of those properties.

Researcher node properties are mostly self-explanatory, e.g. “bio” contains the researcher’s self-reported biography. A few properties are merit explanations. The Researcher node contains a “researchAreasString” and “researchAreas” property. Both properties contain the same information, but “researchAreas” is an array, and the other is a string. The array format is convenient for quick experiments, whereas the string property creates more accurate embeddings since the areas are provided as a single text instead of a list of isolated elements.

The “researcher/bio EmbeddingLARGE” properties are also merit explanations. Each research node contains both a “researcher/bio EmbeddingLARGE” and “SMALL” property.

Both versions are floating point arrays, but the SMALL vector dimensions (1536) are less than the LARGE vector dimensions (3072). Our final model only uses the LARGE version for reasons discussed in the *Embeddings Creation* section.

The Query node is unique in the sense that it only interacts with the Researcher node. It only has two properties: a name and a "queryEmbedding". The "queryEmbedding" is an embedding of some research interest. Our query nodes vary from the more semantically abstract "Deep Learning" to well-defined "Neuroscience." We used these nodes to semantically compare which research/bio embeddings were most like the query embedding. Without the query nodes we could only group researchers based on similarities with other researchers; we could not group researchers on similarities with research fields.

Present in the nodes but not previously mentioned is the "BioAndAreas" property. This property is just a concatenation of biographies and research areas. If one of the areas was missing, we set "BioAndAreas" as whichever one was available, meaning a few nodes have two different properties with the same content. We did this so as not to exclude researchers when "BioAndAreaEmbeddingsLARGE" was used for grouping. This property is variable as a result, but the upshot is that researchers that were missing only one type are included in groupings. See the *Testing Methods* sections for further information.

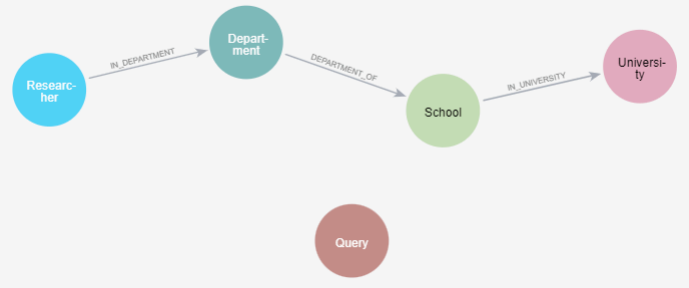

Figure 2: In Cypher pseudocode: (Researcher) is [IN_DEPARTMENT] of (Department). (Department) is a [DEPARTMENT_OF] a (School). The (School) is [IN_UNIVERSITY] of (University). Ex.: (Nurcan Yuruk) is [IN_DEPARTMENT] of (Computer Science) which is a [DEPARTMENT_OF] (Lyle) which is [IN_UNIVERSITY] of (SMU).

Embeddings Creation

Vectors are a list of floating-point numbers. It has become popular in machine learning to use vectors - commonly referred to as word embeddings - to represent words and phrases [3]. We used two word embedding transformations offered by OpenAI: 'text-embedding-3-small' and 'text-embedding-3-large.' The small embeddings have a dimension of 1536 while large embeddings have a dimension of 3072 [4]. Large embeddings encode more semantic nuance relative to small ones, scoring a higher average on the Multi-Language Retrieval Benchmark (54.9%) compared to the small embedding (44%). Large embeddings also beat small embeddings by 2.3% on the Massive Text Embedding Benchmark (MTEB). The tradeoff is that small embeddings cost less to compute and store [5].

Embeddings are great at capturing semantic meaning – not only the word or phrase, but the underlying context behind it. For example, if we embed the word "apple" in a sentence that includes "ate" and "green," "apple" would be closer associated with "fruit" than "iPhone".

Cosine Distance Algorithm

Cosine similarity algorithms judge word similarity by measuring the distance between embeddings [6]. The algorithm ranks each similarity score from highest to lowest and returns the node associated with the score. This process answers the query with the most relevant nodes (e.g. "fruit" and "iPhone" nodes grouped with the queried word "apple").

The two most common ways to gauge similarity between embeddings are distance and angle, or Euclidean and Cosine distance respectively. Euclidean distance uses vector magnitude to calculate how far one node is from another. Cosine distance uses the dot product to calculate similarity [7]. We tested whether the distance metrics yielded different results for the same queries and found none. We used cosine similarity because embeddings are slightly faster to compute [8]. The scores computed with Cosine distance were roughly 1.5 times greater than those computed with Euclidean distance. However, because both metrics yielded the same researchers and the same relative ordering, the actual scores are arbitrary.

Testing Methods

We tested the following three node properties: "bio," "researchAreas", and "BioAndAreas". We wanted to see which combination yielded the most accurate groupings. We asked two researchers familiar with their department to verify rankings. Biographies capture a researcher's educational and professional background and the research they have conducted, while the research areas identify the specific fields a researcher engages in. BioAndAreas concatenates the previous properties with the idea that more information is better.

We used various queries to group researchers. The "name" query answers which 10 researchers are like a specific researcher. We compared the properties of the 10 researchers against the specific researcher's properties to validate if the grouping was accurate.

The "key-term" query answers which 10 researchers used a certain phrase in their biography or research area. Querying the phrase "machine learning" groups researchers who use that exact phrase in their properties. This method is the same as the previous one, except it can find multiple researchers who fit the description (there can only be one researcher but multiple researchers who study machine learning).

The obvious limitation with the previous query is that it misses researchers in that field who do not use the exact term. We created special Query nodes to bypass the limitation. As mentioned in the *Schematic of our Graph*, the query nodes contain an embedding of a certain research field, such as deep learning. We compare the query node embedding to researcher embedded properties to find researchers who are most involved in the field the query node represents. The query-based search is more flexible and associative – this search can find nodes that describe "machine learning" without using the exact term.

Integration with OpenAI LLM

After gathering our data and storing it in a Neo4j graph database, we used the Langchain framework to connect to OpenAI's GPT 3.5 model and implement a RAG system. This framework is accessible via a Python library called Langchain, and it simplifies building LLM applications by providing abstracted objects, functions, and operators. We made use of the library's chain, retriever, and agent features.

Chains

Chains are objects that contain a "sequence of calls" [10], where the output of one process is passed as input to the next. Chains are primarily defined using Langchain Expression Language (LCEL), and they can contain functions, LLM prompts, document retrievers, as well as LLM objects and agent executor objects.

Retrievers

Retriever objects serve as wrappers for vector stores. Vector stores are databases that are used to store and manage high-dimensional vector data. The retriever objects can accept a text-formatted query, embed it, and run a similarity search by comparing the query embedding against those in the vector store [11]. Upon identifying the documents with embeddings most like the query, the retriever will return the text data associated with the documents. In this project, we used retrievers to wrap Neo4jVector objects, which access our graph database to compare vector embeddings and retrieve text data from up to six relevant nodes.

Agents and Tools

Langchain documentation describes agents as "systems that use LLMs as reasoning engines to determine which actions to take and the inputs to pass them" [12]. As of Langchain version 0.2, agent objects can be initialized using an LLM API connection, an LLM prompt, and a list of tools available to the agent. The agent is then wrapped with an agent executor class, which can be invoked directly or called as part of a chain.

Broadly, tools are "interfaces that an agent may use to interact with the world" [13]. This includes interacting with other applications, such as searching the internet or accessing a database, but can include a myriad of other actions that inform the agent's final text output. Each tool is initialized with a name, a description, and a function that will be called upon the tool's execution. Both the name and description are represented as text data that an agent will use to decide whether it should use the tool or not. As such, these variables can be designed to include specific instructions and examples that tailor the agent's choice of tools and achieve results closer to ones that are desired or expected.

RAG Architecture

The major components of our RAG system can be split between its integration with OpenAI and Neo4j, and how it employs Langchain features to handle various query types.

Connections to OpenAI and Neo4j

The first section of our RAG implementation concerns establishing connections to OpenAI and the Neo4j database and initializing the retrievers that perform a search task. Langchain provides multiple classes that streamline connecting to various databases and language models, including OpenAI and Neo4j.

We initialized our OpenAI connection using a ChatOpenAI object, where we provided an API key, selected the GPT 3.5 model, and set the model temperature to zero. The temperature of a language model indicates the amount of randomness applied while generating its responses. Since the goal of RAG is to produce grounded, factual answers, setting the OpenAI model's temperature to zero improves its accuracy and prevents hallucination. Additionally, using the GPT 3.5 model during the development stage allowed us to test our RAG system using a faster and less resource-intensive LLM. Each of these settings are saved in the ChatOpenAI object, making it convenient to invoke the LLM or change its settings at a later time.

The Langchain library also provides various classes that can access Neo4j graph databases. We initialized our graph connection by storing the graph url and authentication information in a Neo4jGraph object. The graph object allowed us to run Cypher queries on the graph from the application

code, and to grant graph access to other objects by passing it as a parameter. We initialized two Neo4jVector objects to access the "researchEmbeddingLARGE" and "bioEmbeddingLARGE" data stored in our database. Finally, we used Langchain's retriever interface to wrap the Neo4jVector objects and set their standard number of returned documents to six. While vector stores can also perform similarity searches, wrapping them as retrievers fixes extraneous function parameters so that the different queries that run with the same retriever object are executed with comparable settings.

Query Handling

To handle various types of natural language queries, we used a Langchain agent to categorize queries based on whether they provide name-based information or field-based information. Examples of these query types are available in the results section, where queries 1-2 are name-based, and queries 3-5 are field-based.

Upon receiving a field-based query, the agent calls a tool that accepts string input. This input is formatted by extracting the research area descriptions from the query, separating them by commas if there are multiple, and then passing them into the tool. The tool then calls a function to perform a retriever-based search with the input, and formats the results to extract the researchAreasString, name, and department text data from returned documents. In this action, the tool's input format removes extraneous words from the query, thereby refining the vector embedding search.

Since name-based queries are highly useful, but do not provide any inherent information about research areas, they must be handled separately. Upon receiving a name-based query, the agent calls a custom tool that identifies a specific researcher in the graph and runs a similarity search using their text data. This tool accepts the original query string as input, and it consists of an LLM call, two functions, and a retriever call. The LLM call compares the query with a list of names that exist in the graph and extracts the in-graph name that most closely matches the query. This name is passed to a function that queries the graph to retrieve the particular researcher's researchAreasString property. Finally, the researchAreasString property is passed into a retriever call, and the output is formatted before returning the search results to the agent. This process closely mirrors the search steps performed by the database queries, where a specific node is identified, and a similarity search is performed using its existing properties.

III. Results

Query Examples

To evaluate the performance of our finalized RAG model, we designed five queries that would test its ability to identify researchers who specialize in fields relevant to a user's question. These test queries are as follows:

1. Which professors have similar research interests as Eric Larson?
2. Which professors have similar research interests as Mehak Gupta?
3. Which professors have deep learning as a research interest?
4. Which professors have natural language processing as a research interest?
5. Which professors have neuroscience as a research interest?

We designed these queries with an emphasis on Computer Science (CS) researchers and interdisciplinary fields. Since we are most familiar with research areas in CS, focusing on CS faculty members improves our ability to verify the quality of our search results by comparing the lists of research areas associated with each professor. As such, we selected the CS department researchers Eric Larson, Mehak Gupta, and Nurcan Yuruk as query subjects. We designed the field-based queries according to similar logic, with an increased focus on interdisciplinary fields. Deep learning and natural language processing are primarily CS fields, but their connection to the subject of artificial intelligence makes them strong candidates for interdisciplinary research in fields such as mathematics, statistics, and cognitive science. Neuroscience is also highly interdisciplinary, as it combines fields including biology, psychology, and even computer science to study the human nervous system. The multidisciplinary nature of these fields allows our queries to demonstrate the RAG model's ability to identify similar research areas across seemingly disparate departments.

In the following sections, results are discussed individually for each of the above queries. Research area text data is provided on a case-by-case basis, and a comprehensive list of research areas for the faculty members noted in the results section may be viewed in Appendix II.

Similar Research as Eric Larson

The first query produced similar results from the Cypher queries and RAG model. There is a 4/5 similarity between the two sets of results, as both search techniques identified Corey Clark, Mehak Gupta, Frank Coyle, and Michael Hahsler as having similar research interests as Eric Larson. The two unique results were Scott Douglas, identified by the Cypher query, and Klyne Smith, identified by the RAG model. Since

Scott Douglas is part of the Electrical and Computer Engineering department, the Cypher query performed slightly better in identifying faculty members from different disciplines. However, the 4/5 similarity between the two techniques indicates that the RAG model produces a comparable quality of results relative to a direct database query.

Table 3: Q1 Cypher and RAG Results

Cypher Query Output			RAG Retriever Matches	
Names	**Department**	**Score**	**Name**	**Department**
Corey Clark	Computer Science	0.8294	Corey Clark	Computer Science
Frank Coyle	Computer Science	0.8115	Mehak Gupta	Computer Science
Michael Hahsler	Computer Science	0.8071	Frank Coyle	Computer Science
Scott Douglas	Electrical Computer Engineering	0.7993	Michael Hahsler	Computer Science
Mehak Gupta	Computer Science	0.7825	Klyne Smith	Computer Science

RAG Text Output: "Professors with similar research interests as Eric Larson include Corey Clark, Mehak Gupta, Frank Coyle, Michael Hahsler, and Klyne Smith."

Similar Research as Mehak Gupta

The second query produced a 3/5 similarity between the Cypher query results and RAG model output, with the RAG model performing slightly better in accuracy and in identifying researchers from separate departments. Both search methods identified Labiba Jahan, Michael Hahsler, and Eric Larson as having similar research interests to Mehak Gupta. However, while the Cypher query also identified Kasilingam Periyasamy and Nurcan Yuruk, the RAG model instead identified the professors Jing Cao from the Statistics and Data Science department, and David (King Ip) Lin from the CS department.

Table 4: Q2 Cypher and RAG Results

Cypher Query Output			RAG Retriever Matches	
Names	**Department**	**Score**	**Name**	**Department**
Labiba Jahan	Computer Science	0.8294	Labiba Jahan	Computer Science
Kasilingam Periyasamy	Computer Science	0.8115	Eric Larson	Computer Science
Michael Hahsler	Computer Science	0.8071	Jing Cao	Statistics and Data Science
Eric Larson	Computer Science	0.7993	Michael Hahsler	Computer Science
Nurcan Yuruk	Computer Science	0.7825	David (King Ip) Lin	Computer Science

RAG Text Output: "Labiba Jahan and David (King Ip) Lin are professors with research interests like Mehak Gupta."

Each of the four unique search results contain fields relevant to Mehak Gupta's research interests. However, the RAG model's results are both more interdisciplinary and slightly more relevant. The RAG model identified Jing Cao from the Statistics and Data Science department, and David (King Ip) Lin from the CS department. Since Mehak Gupta is from the CS department, Jing Cao is a strong interdisciplinary result. Additionally, they both have research interests in "machine learning," as well as similar interests in "medical data" and "clinical trial design". David (King Ip) Lin is also a strong match, as he shares the "natural language processing" interest with Mehak Gupta. While Nurcan Yuruk and Kasilingam Periyasamy are valid results, as their respective interests in "healthcare applications" and "applied machine learning" align with Mehak Gupta's research areas, these similarities are less specific than those identified by the RAG model's retriever output. Overall, both the Cypher and RAG searches produced accurate results for Q2, with the RAG model's output being slightly more relevant.

Table 5: Q2 Research Areas Comparison

Names	Research Areas
Mehak Gupta	Deep Learning, Supervised and Semi-supervised machine learning, Structured and Unstructured medical data, Predictive modeling, Natural Language Processing
Kasilingam Periyasamy	Software Engineering, Formal Methods, Data Science, Healthcare Applications
Nurcan Yuruk	Data Mining, Social Network Analysis, Applied Machine Learning
Jing Cao	Clinical trial design and analysis, Bayesian methods and applications, Machine learning and text mining
David (King Ip) Lin	Natural Language Processing, Database Systems, Data mining

Research in the Deep Learning Field

The Cypher query search and RAG model performed similarly well in identifying researchers in the Deep Learning field. Of the 6 researchers identified by the RAG model, all 6 were also identified in the Cypher query's 10 results, with a 5/6 similarity between the top 6 results from each search. In these top 6, the Cypher query identified 4 researchers from the CS department, and 2 from outside the CS department, while the RAG model identified 5 researchers from the CS department, and 1 from a separate department. In particular, the Mathematics department researcher Difeng Cai was matched by both the Cypher query and the RAG model. His research areas are an accurate interdisciplinary match, as they specifically include "Studying mathematical structures in deep learning models for scalable computation". Altogether, both the Cypher query and RAG model found relevant and interdisciplinary results for Q4.

The additional four results from the Cypher query further demonstrate the ability of embedding-based similarity search to identify interdisciplinary results. In total, the database query matched 6 CS department researchers, 2 Mathematics department researchers, and 2 researchers from the Operations Research and Engineering Management department.

Table 6: Similarity Search Table for Deep Learning

Cypher Query Output			**RAG Retriever Matches**	
Names	Department	Score	Name	Department
Mehak Gupta	Computer Science	0.7232	Mehak Gupta	Computer Science
Eric Larson	Computer Science	0.7140	Eric Larson	Computer Science
Frank Coyle	Computer Science	0.6980	Frank Coyle	Computer Science
Difeng Cai	Mathematics	0.6954	Michael Hahsler	Computer Science
Labiba Jahan	Computer Science	0.6854	Labiba Jahan	Computer Science
Miju Ahn	Operations Research and Engineering Management	0.6789	Difeng Cai	Mathematics
Wei Cai	Mathematics	0.6773	--	--
Corey Clark	Computer Science	0.6743	--	--
Digvijay Boob	Operations Research and Engineering Management	0.6722	--	--

Research in Natural Language Processing

The results between the Cypher query and the RAG output have an 8/10 similarity. Both identified David (King Ip) Lin, Labiba Jahan, Frank Coyle, Mehak Gupta, Joshua Oltmanns, Scott Douglas, Michael Hahsler, and Nurcan Yuruk. Joshua Oltmanns, a professor in the Psychology department, is a good interdisciplinary result here since the majority of the professors output primarily study engineering, or more specifically computer science. This shows the diversity of the RAG model and Cypher query's ability to find relevant matches across different fields of study.

Table 7: Q5 Cypher and RAG Results

Cypher Query Output			**RAG Retriever Matches**	
Names	**Department**	**Score (rounded to fourth decimal)**	**Name**	**Department**
David (King Ip) Lin	Computer Science	0.7232	Labiba Jahan	Computer Science
Labiba Jahan	Computer Science	0.7140	David (King Ip) Lin	Computer Science
Frank Coyle	Computer Science	0.6980	Frank Coyle	Computer Science
Sarah Kucker	Psychology	0.6954	Michael Hahsler	Computer Science
Mehak Gupta	Computer Science	0.6854	Mehak Gupta	Computer Science
Joshua Oltmanns	Psychology	0.6789	Nurcan Yuruk	Computer Science
Scott Douglas	Electrical Computer Engineering	0.6773	Joshua Oltmanns	Psychology
Eric Larson	Computer Science	0.6743	Scott Douglas	Electrical Computer Engineering
Michael Hahsler	Computer Science	0.6722	Eric Larson	Computer Science
Nurcan Yuruk	Computer Science	0.6715	Sarah Kucker	Psychology

RAG Text Output: "Professors at SMU who have natural language processing as a research interest include Labiba Jahan, David (King Ip) Lin, Frank Coyle, Michael Hahsler, Mehak Gupta, Nurcan Yuruk, Joshua Oltmanns, and Eric Larson."

The results between the query node for Natural Language Processing (NLP) and the query node for Natural Language Processing with a focus on Computer Science and Machine Learning have a 9/10 similarity. Despite the results being nearly identical, the results dramatically shift between the two. For the NLP node, the top 7 researchers are diverse with 4 of them in the computer science department, 2 in the psychology department, and 1 in electrical computer engineering. Meanwhile, for the NLP node with focus, all the top 7 researchers are in the computer science department. Joshua Oltmanns was previously in the top 7 researchers for the NLP query, but he lands in last for the top 10 researchers for the NLP with focus query. This is most likely because the NLP with focus node emphasizes "computer science" leading to less diverse results.

Table 8: NLP vs. NLP with a focus on CS & ML

Query Node: Natural Language Processing			**Query Node: NLP with a focus on Computer Science and Machine Learning**		
Names	**Department**	**Score**	**Names**	**Department**	**Score**
David (King Ip) Lin	Computer Science	0.7232	Labiba Jahan	Computer Science	0.7711
Labiba Jahan	Computer Science	0.7140	Frank Coyle	Computer Science	0.7691
Frank Coyle	Computer Science	0.6980	David (King Ip) Lin	Computer Science	0.7621
Sarah Kucker	Psychology	0.6954	Eric Larson	Computer Science	0.7273
Mehak Gupta	Computer Science	0.6854	Mehak Gupta	Computer Science	0.7244
Joshua Oltmanns	Psychology	0.6789	Michael Hahsler	Computer Science	0.7118
Scott Douglas	Electrical Computer Engineering	0.6773	Nurcan Yuruk	Computer Science	0.7063
Eric Larson	Computer Science	0.6743	Corey Clark	Computer Science	0.7054
--	--	--	Scott Douglas	Electrical Computer Engineering	0.7031
--	--	--	Joshua Oltmanns	Psychology	0.6978

Research in the Neuroscience field

The Cypher query and RAG model searches produced highly similar outputs for Q6. The top 6 results for each search have a 5/6 similarity, and all 6 researchers identified by the RAG model were also matched in the 10 results from the Cypher query. The top 6 RAG retriever results included 3 mathematics department researchers, 2 biological science department researchers, and 1 in the electrical computer engineering department. The Cypher query resulted in 3 researchers in mathematics, 3 in psychology, 2 in biological sciences, 2 in electrical computer engineering. As such, both searches display capabilities of identifying similar research across separate departments, with strong consistency in results between the different search methods.

The Cypher query results for Holly Bowen and Prasanna Rangarajan both demonstrate the effectiveness of using embedding-based semantic search to identify similar research areas. Holly Bowen's research areas include "Memory, Emotion, Motivation, Aging" and "fMRI". Although her research areas do not include the phrase "neuroscience", she was still matched by the embedding search. This demonstrates the semantic search's ability to detect similarities based on contextual meaning, which gives both of our search methods greater flexibility relative to traditional keyword search models. Likewise, Prasanna Rangarajan's research areas do not include the keyword "neuroscience". However, her interests in "Neuromorphic Sensing, Optical Engineering", and "Computer Vision", each relate to computational and engineering aspects of the neuroscience field. By comparing research fields based on their context and semantic meaning, the Q6 Cypher query identified meaningful interdisciplinary similarities that would have been overlooked by a keyword search. Although the RAG model's results did not include Holly Bowen or Prasanna Rangarajan, it is likely that they would appear in a larger batch of retrieval results, given the demonstrated consistency between the Cypher query and RAG output for the neuroscience focused query.

Table 9: Research Results for Neuroscience Field

Cypher Query Output			**RAG Retriever Matches**	
Names	**Department**	**Score**	**Name**	**Department**
Leslie-Ann Asmus	Mathematics	0.7154	Leslie-Ann Asmus	Mathematics
Zhihao Wu	Biological Sciences	0.7101	Kathryn Hedrick	Mathematics
Kathyrn Hedrick	Mathematics	0.7049	Zhihao Wu	Biological Sciences
Andrea Barreiro	Mathematics	0.6918	Andrea Barreiro	Mathematics
Amy L. Brewster	Biological Sciences	0.6871	Amy L. Brester	Biological Sciences
Holly Bowen	Psychology	0.6781	Carlos E. Davila	Electrical Computer Engineering

Carlos E. Davila	Electrical Computer Engineering	0.6723	--	--
Stephanie Wilson	Psychology	0.6501	--	--
Sarah Kucker	Psychology	0.6408	--	--
Prasanna Rangarajan	Electrical Computer Engineering	0.6390	--	--

Real Life Application

Inter-departmental Collaborations

The developed system significantly enhances inter-departmental collaborations by identifying researchers with similar or complementary research interests across different departments. For example, a researcher in Computer Science may discover potential collaborators in the Psychology department who are investigating computational models of cognition. Utilizing this tool to identify these connections enables universities to come up with more interdisciplinary research projects.

Research Networking

Researchers often face challenges in identifying and connecting with colleagues outside their immediate professional circles. Our system provides an efficient method to locate and connect with researchers who share similar interests, even if they belong to different departments or schools. The system allows researchers to rapidly build a network of potential collaborators.

Grant Applications

Forming a diverse team with diverse backgrounds can substantially improve the chances of success in research grant applications. This system aids in identifying potential team members who can contribute different perspectives and expertise; therefore, strengthening the grant proposal. Also, it can be used to demonstrate the interdisciplinary nature of the proposed research, which is a key requirement for many funding agencies.

Academic Events and Conferences

Organizers of academic events and conferences can use this system to identify potential speakers or panelists with knowledge in specific areas. It can also help with forming discussion groups or workshops that bring together researchers from different disciplines to discuss common challenges.

IV. Conclusion

This paper demonstrates the development of a natural language interface for a knowledge graph database using OpenAI's ChatGPT-3.5 model. The model is integrated with a Neo4j database via the Langchain framework. This user-friendly system allows non-experts to query research data intuitively, addressing a common issue in database accessibility for users.

We evaluated the system's performance through five test queries that successfully identified professors with similar research interests. In the interdisciplinary research areas, our model demonstrates high accuracy and efficiency, identifying strong interdisciplinary connections that traditional keyword searches could not.

The natural language interface provides a way for researchers to query the database in English as opposed to learning coding languages to directly query the database from the backend. The system holds significant potential to enhance research networking, improve the formation of interdisciplinary teams for grant applications, and assist in organizing academic events and conferences. This approach not only improves accessibility and usability but also fosters greater collaboration and innovation within the academic community. Future steps will focus on improving system accuracy by considering researcher publications as a metric, expanding capabilities, adding more researchers from different universities, and exploring broader applications.

References

[1] Zhao, Wayne et al. "A Survey of Large Language Models." *Arxiv*, 11 Mar. 2025, https://arxiv.org/abs/2303.18223.

[2] Neo4j, "Graph Thinking," *Neo4j,* https://graphacademy.neo4j.com/courses/neo4j-fundamentals/1-graph-thinking/ (accessed Jul. 29, 2024)

[3] Barnard, Joel. "What are Word Embeddings?" *IBM,* https://www.ibm.com/think/topics/word-embeddings#:~:text=Word%20embeddings%20have%20proven%20invaluable,way%20compared%20to%20traditional%20methods.

[4] OpenAI, "OpenAI API Embeddings," *OpenAI,* https://platform.openai.com/docs/guides/embeddings

[5] OpenAI, "OpenAI Platform Embeddings," *OpenAI,* https://platform.openai.com/docs/guides/embeddings/

[6] Neo4j, "Semantic Indexes - Cypher Manual," *Neo4j,* https://neo4j.com/docs/Cypher-manual/current/indexes/semantic-indexes/overview/ (accessed Jul. 30, 2024).

[7] Neo4j, "Vector Indexes - Cypher Manual," *Neo4j,* https://neo4j.com/docs/Cypher-manual/current/indexes/semantic-indexes/vector-indexes/#similarity-functions (accessed Jul. 30, 2024).

[8] OpenAI, "OpenAI Embeddings FAQ," *OpenAI,* https://platform.openai.com/docs/guides/embeddings

[9] Neo4j, "Procedures - Operations Manual," *Neo4j Graph Data, Platform*. https://neo4j.com/docs/operations-manual/5/reference/procedures/#procedure_db_index_vector_queryNodes (accessed Jul. 30, 2024).

[10] Langchain, "Chains | Langchain," *LangChain,* https://Python.Langchain.com/v0.1/docs/modules/chains/

[11] Langchain, "Retrievers | Langchain," *LangChain,* https://Python.Langchain.com/v0.1/docs/modules/data_connection/retrievers/

[12] Langchain, "Build an Agent | Langchain," *LangChain,* https://Python.Langchain.com/v0.2/docs/tutorials/agents/#:~:text=A%20big%20use%20case%20for (accessed Jul. 30, 2024).

[13] Langchain, "Tools | Langchain," *LangChain,* https://js.Langchain.com/v0.1/docs/modules/agents/tools/#:~:text=Tools%20are%20interfaces%20that%20an (accessed Jul. 30, 2024).

Appendix

Appendix A- I: Node Properties

Node Type	Properties	Type	Relations
Researcher	researchAreasString researchAreas researchEmbeddingLARGE researchEmbeddingSMALL bio bioEmbeddingLARGE bioEmbeddingSMALL BioAndArea BioAndAreaEmbeddingLARGE departmentId name new	STRING LIST LIST LIST STRING LIST LIST STRING LIST LIST STRING STRING BOOLEAN	IN_DEPARTMENT
Department	name	STRING	DEPARTMENT_OF
School	name	STRING	IN_UNIVERSITY
University	name	STRING	None
Query	name queryEmbedding	STRING LIST	None

Appendix A- II: Research Areas Text Data

Researcher	Research Areas
Miju Ahn	Mathematical optimization, Statistical learning, Nonconvex programming
Leslie-Ann Asmus	Mathematical and computational neuroscience, The study of dynamical, information theoretic and computational properties of neural networks, Fluid dynamics
Andrea Barreiro	Mathematical and computational neuroscience, including the study of dynamical, information theoretic and computational properties of neural networks, Fluid dynamics
Digvijay Boob	Design and Analysis of Algorithms, First-order methods, Variational Inequality and Minmax problems, Data-driven (stochastic) optimization, Machine Learning Theory
Holly Bowen	Memory, Emotion, Motivation, Aging, fMRI
Amy L. Brewster	Identify potential therapeutic targets for the prevention, treatment, and control of epilepsy, Identify whether neuro-immune interactions contribute to the construction of hyperexcitable neuronal networks that may promote seizures and cognitive deficits in epilepsy, Neuroscience, Inflammation and Immunity
Difeng Cai	Efficient algorithms for numerical linear algebra, partial differential equations and integral equations, Studying mathematical structures in deep learning models for scalable computation
Wei Cai	Developing advanced fast machine learning, stochastic, and deterministic numerical methods to understand complex physical phenomena with a special focus on quantum phenomena, electromagnetic processes, and wave scattering in random media
Jing Cao	Clinical trial design and analysis, Bayesian methods and applications, Machine learning and text mining
Corey Clark	Human-in-the-loop Machine Learning, Video Games and Artificial Intelligence, Distributed and Privacy Computing, Human and Machine Collaboration, Bias and Explainability in Artificial Intelligence, Game Based Education
Frank Coyle	Machine Learning / Artificial Intelligence, Software Engineering, Large Language Models
Carlos E. Davila	Biomedical and statistical signal processing, Brain-computer interfacing, Computational neuroscience
Scott Douglas	Audio, Array Processing, Machine Learning, Remote Sensing, Speech Enhancement
Maya El Dayeh	Cloud Computing, Data Mining, Software Engineering
Mehak Gupta	Deep Learning, Supervised and Semi-supervised machine learning, Structured and Unstructured medical data, Predictive modeling, Natural Language Processing
Michael Hahsler	Artificial Intelligence, Machine Learning, Data Mining, Data Science
Kathryn Hedrick	Scientific computation, Neuroscience
Labiba Jahan	Natural Language Processing, Narrative Understanding, Machine Learning
Sarah Kucker	Cognitive Development, Language Acquisition, Digital Media, Word Learning, Categorization
Eric Larson	Machine Learning and Deep Learning, Privacy and Security, Applied Machine Learning in Health and Education, Human Machine Teaming and Biometrics
David (King Ip) Lin	Natural Language Processing, Database Systems, Data mining

Chul Moon	Topological Data Analysis, fMRI Data Analysis, Network Analysis, Geophysical Statistics, Forensic Statistics
Joshua Oltmanns	Clinical and Personality Assessment, Clinical Psychology, Psychometrics, Structural Equation Modeling, Natural Language Processing, Artificial Intelligence
Kasilingam Periyasamy	Software Engineering, Formal Methods, Data Science, Healthcare Applications
Prasanna Rangarajan	Computational Imaging, Neuromorphic Sensing, Optical Engineering, Photonics, Computer Vision
Klyne Smith	Software Engineering Life Cycle Model Delivery Engagement, Project Management Optimization and Risk Management, Intersection and Dependencies of

	Software Engineering and Project Management, Requirements and Software Engineering associated with Cybersecurity, Machine Learning, and Artificial Intelligence
Stephanie Wilson	Family relationships, Social interaction, Psychoneuroimmunology, Adulthood and aging, Autonomic function, Ecological momentary assessment
Zhihao Wu	Neuroscience and Gene Regulation, Gene Regulation
Nurcan Yuruk	Data Mining, Social Network Analysis, Applied Machine Learning
Jia Zhang	Data Science Infrastructure, Machine Learning, Scientific Workflow, Software Engineering, Cloud Computing

Appendix A- III: Department Statistics and Research

Department	# of Researchers	Percentage Total (%)	Researchers	School
Computer Science	20	10.60	https://www.smu.edu/lyle/departments/cs/people/faculty	Lyle
Civil and Environmental Engineering + Sustainability and Development	11	5.85	https://www.smu.edu/lyle/departments/cee/faculty + https://www.smu.edu/lyle/departments/multidisciplinary-programs/ma-in-sustainability-and-development	Lyle
Electrical Computer Engineering	18	9.57	https://www.smu.edu/lyle/departments/ece/people/faculty	Lyle
Mechanical Engineering	11	5.85	https://www.smu.edu/lyle/departments/me/people/faculty	Lyle
Operations Research and Engineering Management	12	6.28	https://www.smu.edu/lyle/departments/orem/people/faculty	Lyle
Design and Innovation	6	3.26	https://www.smu.edu/lyle/departments/multidisciplinary-programs/madi/people	Lyle
Biological Sciences	11	5.85	https://www.smu.edu/dedman/academics/departments/biological-sciences/people	Dedman
Earth Sciences	18	9.57	https://www.smu.edu/dedman/academics/departments/earth-sciences/people	Dedman
Mathematics	23	12.20	https://www.smu.edu/dedman/academics/departments/math/people	Dedman
Physics	13	6.91	https://www.physics.smu.edu/web/people/	Dedman
Statistics and Data Science	14	7.45	https://www.smu.edu/dedman/academics/departments/statistics/people	Dedman
Sociology	11	5.85	https://www.smu.edu/dedman/academics/departments/sociology/people	Dedman

Total number adds up to 188 researchers because some fields share researchers. Percentages may not add up to 100% due to rounding