Large Language Model-driven Data Augmentation for Neural Link Predictors

Arth Akhouri

Master of Science
Data Science
School of Informatics
University of Edinburgh
2024

Abstract

In this modern era, all the technological advancements in the fields of Machine Learning (ML) and Artificial Intelligence (AI) are primarily because of data. The way this knowledge is stored is a critical component behind the meteoric rise of these systems. Knowledge representation involves the systematic structuring and organisation of information so that these systems can benefit from easy access to data they can process and understand. Knowledge Graphs (KGs) and Knowledge Bases (KBs) are essential to this process because they make it easier to map complex relationships between entities, which improves the depth of data interaction and the power of AI systems. This research explores the use of large language models (LLMs) to enhance the effectiveness of neural link prediction models within KGs. Link prediction is vital for discovering missing links between entities in a KG and can directly impact its utility and completeness across various domains, like recommender systems and drug interactions. We do this by using the LLM to augment the data available as triples and enhance the overall information stored in the KG. Many complications arose while directly generating triples from LLMs, so we instead focused on generating triples randomly and scoring them using the LLM using In-Context Learning. An in-depth analysis of the generated triples is carried out to evaluate the information being added to the KG. The addition of augmented triples led to modest improvements in the performance of Knowledge Graph Embedding (KGE) models. This was verified across different LLM models and multiple runnings of the same experiments. This methodology is then tested on an actual Drug Interaction KG, and the outputs are analysed. We provide insights into the complex function of data augmentation and emphasise the necessity for careful consideration of context and completeness in KGs. We also illustrate the promise and limitations of employing LLMs for improving KGC datasets and KGE models.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Arth Akhouri*)

Acknowledgements

I want to express my heartfelt gratitude to my supervisor, Dr. Pasquale Minervini, for his unwavering support throughout this dissertation and his consistently helpful attitude. His guidance and insights have been invaluable, and I am deeply appreciative of his commitment to my work. This dissertation would not have been possible without his expert advice and encouragement.

I sincerely want to thank all my friends who have stuck by my side through thick and thin and helped me through all the challenges that I have faced.

Lastly, I would not be here without the unwavering support of my family, especially my parents, who encouraged me every step of the way and stood as steadfast pillars of strength throughout this phase of my life.

Table of Contents

Chapter 1

Introduction

In today's age, every aspect of our lives is influenced by technology, be it as simple as a switch turning on a light or the smartphones that we have in our pockets. Behind every single one of these things is data. It serves as the backbone of modern machine learning (ML) and artificial intelligence (AI) systems in this modern era of technology. They all leverage vast amounts of data (Big Data) to train algorithms that can recognise patterns, make predictions, and perform complex tasks with increasing accuracy. This new push has transformed numerous industries to edge forward by helping them automate decisions and gain new previously unattainable insights, which have helped them make intelligent business decisions. A crucial factor behind this success is how effectively the data is managed, organised and represented.

Knowledge representation is a fundamental aspect of this new technology, and it includes how the information is systematically structured and organised in a system for a machine to understand, process and evaluate it. This often consists of a variety of techniques and models that are used to represent knowledge in order to feed into these advanced systems. The data is evaluated, and a formal structure is created for concepts, relationships and rules that define how entities interact within a particular domain. This knowledge representation serves as a foundation upon which all modern AI applications are built, enabling a more in-depth interaction with the vast amounts of data that power modern technological breakthroughs.

Knowledge representation heavily relies on Knowledge Bases (KB) and Knowledge Graphs (KG) for organised and systemic interlinking of the data. Knowledge Graphs, in particular, serve as an intricate web that precisely captures and maps the complex relationships between various entities, thereby enabling a more nuanced understanding of data. Knowledge bases are relatively more specific, only housing data related to a

particular domain working in tandem with KGs, enhancing their overall structure and comprehensiveness. A massive part of the KGs is the link prediction between its entities. Link Prediction is the task where a system is able to infer missing or new links within the entities of the knowledge graph by analysing the information already available in it. This directly improves the graphs' utility and completeness. Recommender systems, semantic search, and knowledge discovery are all fields that can benefit from uncovering hidden relationships. Predicting possible connections, for example, can improve user experience in recommendation systems by recommending relevant things. It can aid in the discovery of novel gene-protein interactions in biomedical research. It is possible to anticipate future friendships or business relationships using social network analysis.

1.1 The Goal

The traditional link predictor methods rely on probabilistic and heuristic approaches to find missing links in the data. A step up in that direction is with the help of neural link predictors, which harness the power of Machine & Deep Learning (DL) to more efficiently parse through the data of the knowledge graph and identify complex patterns and interactions within the data. A plethora of methods exist to predict missing links within a KG, each unique in its way with its respective advantages and drawbacks. Recent advancements have been made using methods of Complex Query Decomposition, which uses the neural link predictors on decomposed atomic queries to figure out the big picture answers (Arakelyan et al., 2021). This study aims to push in the same direction and explore other link prediction techniques to improve the State-of-the-art (SOTA) performance of current neural link predictors. We strive to achieve this by augmenting the existing data in the Knowledge Graph (KG), specifically by creating and adding new meaningful triples that can optimise link prediction within the KG. The goal of this comprehensive strategy is to increase prediction accuracy while addressing the challenging issues related to missing connection inference in intricate graphs. We hope to advance the field of AI-driven applications while simultaneously enhancing the predictive potential of KGs by utilising the massive computational power of Large Language Models.

Large Language Models (LLMs) represent the recent SOTA models, which are trained on vast amounts of textual data and excel in understanding, generating and processing natural language with human-like proficiency. LLMs have already shown the world their versatility and their usefulness in a host of different applications. Their

ability to pick up complex patterns and subtle semantics from a variety of datasets makes them indispensable for improving a wide range of AI applications. Building on the versatility of LLMs, they offer compelling opportunities to boost the link prediction technologies within KGs. Their robustness with different tasks also enables us to use LLMs in various stages of the pipeline. Firstly, an LLM can read and comprehensively understand the given dataset, making connections that don't exist in the KG yet and creating a new augmented dataset which is more comprehensive and complete. The LLMs can also represent the data better, effectively encoding complex relationships of an entity along with important contextual information, which can help with link prediction tasks.

The accuracy and performance of neural link predictors could be enhanced by incorporating LLMs. Neural link predictors are able to forecast possible relationships within KGs and more accurately infer missing connections by utilising the extensive knowledge representations that LLMs have learnt during training. This integration is poised to yield more profound insights into the complex web of relationships embedded in KGs, facilitating more precise decision-making in multiple domains.

1.2 Motivation

This research is heavily influenced by the need to revolutionise drug interaction studies within healthcare. Various drugs often interact with each other, and some of those combinations can be very harmful. Even the best traditional methodologies struggle to capture and fully understand all the possible complex drug interactions. This drawback causes significant gaps which affect treatment efficacy and directly compromise the safety of the patients. Bates et al. (1995) highlight the consequences of adverse drug interactions and prove that it remains a persistent and concerning issue which needs to be tackled using the new technologies available to us. Farrugia et al. (2023) have already showcased how the utilisation of KGs can help our understanding of drug interactions and mitigate the associated risks with it.

The need to continuously integrate new findings and their implications in drug interactions is further validated by the rapid growth of available biomedical data. Hence, we have utilised the power of LLMs, combined with SOTA link prediction techniques, in our quest to improve the accuracy and efficiency of identifying missing links within KGs. All the methods to be researched are thoroughly tested on drug interaction KG to validate our approach. The results of this study have the capacity to revolutionise the

field of medication interaction research and patient care. Improved prediction skills will result in better clinical judgement, eventually benefitting patient safety and treatment results. Additionally, by advancing our knowledge of drug interactions, this research hopes to guide innovation in the pharmaceutical industry and aid in the creation of safer and more efficient drugs. Our effort intends to directly and significantly improve the healthcare ecosystem by solving the current shortcomings in medication interaction research and offering solid, data-driven solutions (Hripcsak and Albers, 2011). This entails lowering the frequency of unfavourable drug responses, enhancing treatment plans, and providing healthcare providers with accurate and valuable information.

1.3 Structure of Dissertation

In the first part of the dissertation, we go through some fundamental concepts required to understand the components of this dissertation. We get a healthy understanding of KGs, link prediction and LLMs. Then, we look at some standardised datasets and evaluation metrics that will be used throughout this work. We then go to understand and process the data that is already available to us before we start implementing LLMs into the pipeline. Focusing on one approach, we look at the different methodologies to augment the data of our base KG and enhance its information. We then run experiments with varying parameters on different KG datasets and LLM models to see how they affect the proposed methodology. We finally tested our methods on an actual drug interaction KG and analysed its outcomes.

Chapter 2

Background

The development of sophisticated data management and machine learning techniques in recent years has dramatically improved our capacity to handle and comprehend complicated datasets. Google created knowledge Graphs (KGs) to enhance the relevance and accuracy of search results by displaying data about entities and their relationships in an organised, graphical way, which is essential to these developments. Information retrieval is now more straightforward and effective because of this breakthrough, which has completely changed the way we query and interact with data. Improved querying, reasoning, and inference capabilities are made possible by KGs' semantic and machine-learnable nature, which makes them invaluable for a variety of applications like browsing, searching, recommendations, and personalisation.

A key feature of KGs is their ability to represent vast amounts of data from different sources, which enables us to deduce new insights and information from it. Using inference techniques, we can find hidden connections and extract implicit knowledge from the graph, improving our understanding and facilitating better decision-making. Sophisticated techniques like Neural Link Predictors have been created to tackle the problem of missing links in KGs. Furthermore, using vast amounts of text data, LLMs have become highly effective tools in natural language processing. LLMs are able to recognise intricate patterns and correlations. SOTA performance can be advanced in a variety of ways by incorporating LLMs into the link prediction process and taking advantage of their rich semantic representations to improve the precision and effectiveness of predictions made inside knowledge graphs.

2.1 Knowledge Graphs

Google introduced knowledge Graphs with the intention of better searching for a particular thing (Singhal, 2012). In today's age, KGs stand as a cornerstone in modern information systems. Their help in the facilitation of the structured representation of knowledge is so practical that it helps both human understanding and machine processing abilities. These were initially introduced to improve the already impressive Google Search Engine [1]. But since then, they have evolved into very sophisticated frameworks and are now available to use across a plethora of domains.

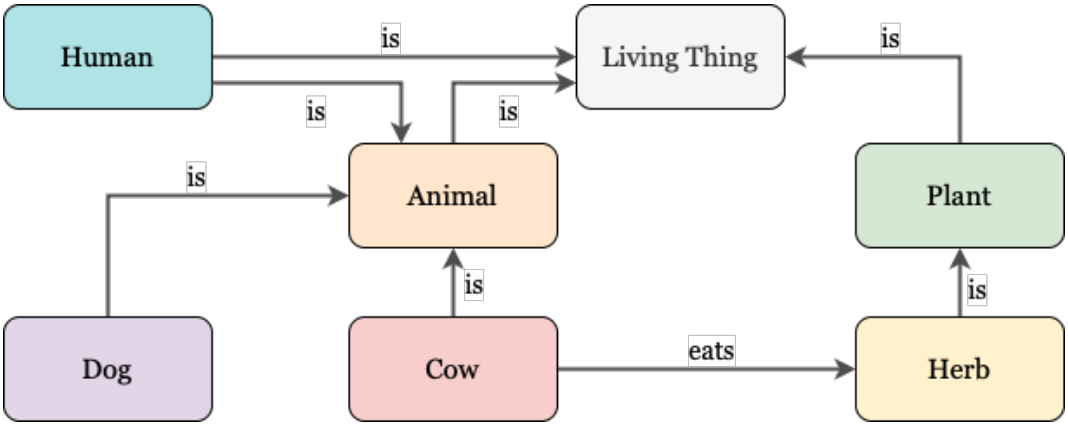


Figure 2.1: A Simple Knowledge Graph

Since its inception, there has been much discussion regarding the formal definition of a KG, most of them missing a particular core while defining them (Ehrlinger and Wöß, 2016). In its most simplistic form, a KG is a massive web of interconnected nodes and edges. Every node here can be connected to multiple other nodes, and the edges between nodes define the relationship between them. The advantage of having this data in a graph-based structure is that it allows the system to capture intricate real-world connections that traditional methods of saving the data might overlook. Ontology, which defines the kinds of entities and relationships, and schema, which details the characteristics and limitations of these things, are two important concepts to take into account while designing knowledge graphs. Because of their rich semantic content, KGs are able to do more than just store data; they can be used to reason and draw conclusions. This makes them indispensable for applications like automated decision-making and natural language comprehension in fields of searching, browsing, recommendations, personalisation and more (Sheth et al., 2019).

[1] https://en.wikipedia.org/wiki/Google_Search

Knowledge Graphs (KGs) are excellent at assisting with data integration tasks because they offer a cohesive framework for integrating data from various heterogeneous sources simultaneously. In today's data-rich environment, when organisations gather enormous volumes of information from many systems and databases at a rapid rate and in different formats, this skill is essential. Because of this graph structure's incredible flexibility, it supports a variety of data types and schemas, entities and relationships which can be represented consistently across formats, independent of their original source. The work of integrating data from several domains or systems is made more effortless by this standard format, allowing for thorough analysis and the development of insights. In order to guarantee that the integrated knowledge stays up-to-date and applicable over time, KGs provide incremental integration of new data sources and dynamic changes. This flexibility is beneficial in fields that are constantly changing and where new information is released on a regular basis, like finance, healthcare, and scientific research.

The growing popularity of Knowledge Graphs (KGs) across several fields has pushed researchers to concentrate on enhancing their creation and application using sophisticated methods and strategies. Automated KG creation from unstructured data sources, including texts, web content, and multimedia sources, is a prominent topic of research. Natural language processing (NLP), machine learning, and information extraction techniques are constantly developing to allow automated procedures that can recognise entities, derive relationships, and add structured knowledge to Knowledge Graphs (KGs).

As the adoption of Knowledge Graphs grows, so too does research into optimising their construction and utilisation. Techniques for automated KG construction from unstructured data, such as text or web content, continue to evolve. Meanwhile, advancements in graph query languages and algorithms enhance the efficiency and scalability of KG-based applications. Developments significantly improve the productivity and scalability of KG-based systems in graph query languages and algorithms. Advanced querying capabilities that navigate intricate interactions within KGs are made possible by graph query languages such as SPARQL Protocol and RDF Query Language (SPARQL) [2], and Cypher. This allows for accurate information retrieval and supports analytical operations. Meanwhile, algorithmic developments concentrate on maximising the performance and scalability of network traversal, pattern matching, and reasoning activities inside of KGs on large-scale datasets. Researchers are paving the road

[2] https://www.ontotext.com/knowledgehub/fundamentals/what-is-sparql/

for more intelligent and flexible knowledge-driven systems that can fully use structured, interconnected data by continuously improving query capabilities and building approaches.

2.2 Link Prediction

The process of finding out the relationship or link between two entities in a network, learning from the already existing relationships and the structure of the network is called link prediction. It is at the heart of graph analysis and network theory with lot of research going on in this field. This is especially advantageous in various domains — it can help find new friends on a social networking site by analysing common interests and shared acquaintances. Another powerful utility of this system is the recommendation system, which can be applied in a plethora of media websites to drive user engagement. This study aims to apply this in the field of drug interactions KGs.

Combining the field of KGs and link prediction, people have discovered that link prediction in within knowledge graphs can help us uncover missing links that were not previously available in the data. These new connections are often not explicitly stated, but derived from other connections in the KG. This process is called inference. The implicit knowledge is extracted and analysed and then the new connections between entities is inferred (Wei et al., 2022). By using inference, we may make use of the graph's hidden patterns to conduct more thorough and precise investigations. This method not only improves data completeness but also yields insightful information which can be very crucial in making important business decisions.

Often, knowledge graphs are massive and very complex with one entity having multiple relations with multiple other entities. Due to this large volume of data, some connections are understandably missed or left unrecorded (Destandau and Fekete, 2021). Here is where the link prediction comes into play, where they can analyse the connections and pattern of the graph and come up with these missing connections.

The implications of this technology in the field of Healthcare is massive. There are various kinds of KGs, a drug interaction KG is a specialised one where all the entities and relationships revolve around different medications or drugs and how they interact with each other and to treat what kind of problems. The relations account for various different properties of each drug, its properties and its effects (Farrugia et al., 2023). By using link prediction techniques, we aim to predict potential interactions between one or more drugs which are currently unavailable in the KG. Analysing a drug's therapeutic

effects, chemical composition, and its known interactions with other medications can help achieve this.

This achievement can help healthcare professionals anticipate new possible side effects, good or bad and can aid the treatment of patients. For research, while developing a new drug, they can check how it will interact with other available chemicals and the link prediction algorithm can help them suggest them to create a more targeted approach which is safer and more effective in treating the problem. By utilising these predictions to assist medical professionals in identifying potential side effects, optimising medication regimens, and ensuring patient safety, healthcare can be made more effective and safe (Espinosa-Bosch et al., 2012). This can also lead to us finding how one particular drug can help in treatment of something else by combining it with some pre-existing medication. The link prediction algorithm can look for similar reactions and recommend medications accordingly.

2.2.1 Similarity-Based Link Prediction Models

Link prediction technologies have been one of the most worked fields in network theory and they have made huge strides over the years. It initially began with simple link predictors which just used to consider the connected entities of a particular entity and used to derive potential links form the same. One of them is the common neighbours approach. It is a simple and intuitive way of finding missing connections which operates under the assumption that entities with some relation to different entities can be connected. It calculates a score on the basis of how many common entities are there between two entities (Newman, 2003). To calculate the common neighbours score CN(u, v), where u and v are two entities, we can simply use:

$$\mathrm{CN}(u,v) = |\Gamma(u) \cap \Gamma(v)| \tag{2.1}$$

where $|\Gamma(u)|$ and $|\Gamma(v)|$ represent all the connections of u and v, respectively.

The problem with this methodology is that it doesn't factor the kind of connection between the entities. Furthermore, it tends to favour entities with higher connections which would not be feasible in a drug interaction KG. This can cause this methodology to overlook possible less likely meaningful connections. It also tends to perform poorly in a KG where there are a lot of connections overlooking some potential weak connections.

The Adamic-Adar index builds on top of the common neighbours approach to be a

more valuable tool in determining potential connections. It assigns higher importance to entities with less connected common neighbours under the assumption that they might be more valuable at a later stage (Adamic and Adar, 2003). To calculate the Adamic-Adar index AA(u, v), where u and v are two entities, we can use:

$$\mathrm{AA}(u,v) = \sum_{w \in \Gamma(u) \cap \Gamma(v)} \frac{1}{\log |\Gamma(w)|} \tag{2.2}$$

where $\Gamma(u)$ and $\Gamma(v)$ denote the sets of neighbours of entities u and v, respectively. $|\Gamma(w)|$ represents the degree (number of neighbours) of entity$w$.

The logarithmic term $\log |\Gamma(w)|$ weighs the entities accordingly and hence is able to diminish the dominance of highly connected entities. It gives a more balanced score which is more effective in predicting links. But this method also struggles when there are entities with similar connections to other entities.

Another improvement on the common neighbours approach is the Jaccard's coefficient. This returns a value between 0 and 1 which quantifies the similarity between the sets of neighbours of two entities within a network (Jaccard, 1901). For two entities u and v in a network, Jaccard's coefficient JC(u, v) is computed as the ratio of the intersection of their neighbour sets to the union of their neighbour sets:

$$\mathrm{JC}(u,v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|} \tag{2.3}$$

where,

- $\Gamma(u)$ and $\Gamma(v)$ represent sets of neighbours (connections) of entities u and v, respectively.
- $|\Gamma(u) \cap \Gamma(v)|$ is the count of common neighbours shared by entities u and v
- $|\Gamma(u) \cup \Gamma(v)|$ is the count of unique neighbours of both entities.

This metric, which normalises the prediction score based on the total neighbourhood sizes of the participating entities, is especially helpful in networks where entities display differing degrees of connectedness. All these simple methodologies while effective, are unable to handle the size of big knowledge graphs. This limitation paved the way for neural link predictors, which harness the power of ML and DL to predict missing links in the graphs. By learning from known connected and unlinked pairings, neural link predictors first encode the network structure and entity attributes into low-dimensional representations and then use those representations to predict missing links based on similarity scores between entities (Nickel et al., 2015).

2.2.2 Machine and Deep Learning Based Link Prediction Models

One of the best methodologies that has come out of using ML and DL in Knowledge Graphs is the Graph Neural Network (GNN). They are specifically designed to operate on graphs and show a remarkable improvement over simple link prediction techniques. They are so well suited for predicting missing links because they are able to leverage the graph's topology and entity features to learn embeddings that encapsulate both the local surroundings and the overall graph structure (Zhou et al., 2020). This enables them to capture complex dependencies and relationships between the entities. Another boost over the traditional methods is that they learn from the data instead and fine tune their parameters for more efficient predictions.

The versatile use of GNNs is also because of their adaptability to different kinds of graphs with knowledge graph being no exception in that. Be it sparse or dense, homogeneous or heterogeneous, GNNs can adapt to various structures, making them versatile across different domains. Despite showing a huge improvement over other methods, they do not come without their drawbacks. Due to the massive size of KGs, it can very computationally expensive to train GNNs on large KGs requiring significant time and resources. Similar to other large models, their explainability is also a challenge. They are sometimes black box models which are difficult to interpret as to how they arrived at a particular output (Xu et al., 2019). A careful analysis and thorough understanding of the structure of the GNN model can help us understand how the model predicts a particular link.

GNNs are taken further by Graph Convolution Networks (GCN), which are a specific type of GNN where they apply convolution operations to graph data. They are very useful as they able to efficiently learn hierarchical relationships between entities again while capturing both local and global structure of the graph (Hamilton et al., 2018). Because of this, they are very useful for tasks requiring accurate modelling of the connections and relationships between nodes. GCNs are durable and computationally efficient, but they have trouble capturing long-range dependencies in deeply layered graphs (Zhang et al., 2019). To balance complexity and performance, careful architectural design and hyperparameter tuning is needed.

Being influenced by the latest models and to push this technology even further, people also introduced Graph Attention networks (GAT), which incorporates the attention mechanism within GNN. The attention mechanism shines through, allowing the network to focus on important entities by learning from the data of the graph and updating its

representation accordingly (Lee et al., 2023). When there are large variations in the relative significance of neighbours, GATs perform exceptionally well because their attention weights may be dynamically adjusted to highlight important relationships. They perform especially well on heterogeneous graphs but again face the challenge of being difficult to train owing to the added attention mechanism as well.

2.3 Datasets

For Knowledge Graph Completion (KGC), there are some fixed datasets which the entire community uses to check the feasibility of their new method. The two most important ones are the variations of WN18 and FB15K (Lacroix et al., 2018). Freebase [3] was a large community developed knowledge base where members could contribute data about various subjects in a structured format. The FB15K dataset originated from freebase and has been used extensively for training and evaluating knowledge graph embedding and knowledge graph completion models (Bordes et al., 2013*a*). But it was soon found that the FB15K dataset had various direct and inverse relations among the training and validation set which allowed the models to learn rather than generalise over the content in it.

Due to this drawback, Toutanova and Chen (2015) created the FB15K-237 datasets to remove the direct and inverse relations from the dataset. This resulting dataset which was created by reducing some samples from the original one provided a more realistic and challenging benchmark for KGC methodologies and models. It contains of 14,541 entities and 237 relations, with 272,115 training triples, 17,535 validation triples and 20,466 test triples.

WordNet is a lexical database of semantic relations between words that links words into semantic relations including synonyms, hyponyms, and meronyms. The synonyms are grouped into synsets with short definitions and usage examples [4]. WordNet has also been used to create a KGC dataset — WN18 (Bordes et al., 2014). But the problem was, it too suffered from the direct and inverse relationships across it train, validation and test sets which caused KGC models to give high metrics on it due to simple memorisation and not actually learning the patterns in the KG.

Similar to the FB15K-237, Dettmers et al. (2018) created the WN18-RR dataset removing all the reversible relations in the dataset. This again has made the dataset

[3] `https://www.failory.com/google/freebase`
[4] `https://en.wikipedia.org/wiki/WordNet`

more reliable to test KGC models true relational learning capabilities and has also made it more challenging for models to learn the data. The dataset consists of 40,943 entities and 11 relations. They are divided in such a way that there are 86,835 training triples, 3,034 validation triples, and 3,134 test triples. Both the FB15K-237 and the WN18-RR have become a critical resource for benchmarking KGC models directly helping the development of new and more sophisticated algorithms.

A relatively new dataset has recently gained some popularity for benchmarking the KGC models. It is the Codex dataset introduced by Safavi and Koutra (2020). This dataset was introduced with the goal of addressing some of the issues in the previous datasets like data quality, the availability of inverse and direct relations and to some extent even the difficulty of task. Codex was specifically created by focusing on more meaningful and challenging triples which ensured high data quality and removed any noise from the data.

It comes in 2 variants — Codex-S and Codex-M. Codex-S is the smaller dataset with 2,034 entities and 42 relations. It consists of 34,928 training triples, 1,993 validation triples, and 1,986 test triples. On the other hand, the Codex-M is a slightly bigger model with 17,050 entities, 51 relations, and has 185,584 training triples, 10,310 validation triples, and 10,311 test triples. This new dataset has proved to be a very good testing point for all KGC models promoting them to be more robust and better handle complex and realistic scenarios.

2.4 Large Language Models

A significant advancement in the field of Natural Language Processing (NLP) is evidenced by the existence and continuous development of Large Language Models (LLMs). They are designed in such a way that they reach new levels of human understanding that were previously unattainable by other dense models. They do so by learning the intricate patterns and relationships from vast amounts of textual data present in the language (Radford and Narasimhan, 2018). This also ensures that the information they generate is contextually relevant and follows language-appropriate grammar.

As most recent developments, they have also been derived from complex deep neural network architectures which train a lot of parameters to fine-tune their output. Initially, these networks were used for the task of NLP and image recognition, where they were used to learn abstract features and capture complex relationships in the data. However, as the tasks grew more complex, they even failed to fully understand the data

and missed the underlying patterns required to process the output task at hand.

This is when LLMs powered by the Transformer came into the picture. In their 2017 paper "Attention is All You Need", Vaswani et al. (2017) introduced the mechanism of self-attention, which allowed the model to differ their weights to focus on the important words of a sentence relative to each other. This drastically improved a model's ability to capture the context in a given text more effectively and use it to understand the data better. This also enabled the model to process the entire input sequence at once instead of sequentially. Stacking multiple layers of the self-attention layers and complex feed-forward networks led to the formation of the Transformer block, which allows it to learn hierarchical representations of language.

For training an LLM, one often requires a large amount of textual data, which generally consists of billions of words across thousands of documents and a variety of different sources like books, articles, websites and more. This is how a model is able to predict the next word in a sentence while being fully aware of the context of the sentence and the topic at hand. It also ensures that the model gradually builds up its understanding of grammar, syntax, semantics and even some degree of knowledge of that specific topic (Naveed et al., 2024). As the name suggests, these models are massive, with now millions and billions of parameters, all of which are trained for quite a bit for them to understand and capture the subtle nuance and intricate patterns in the language.

All the perks of LLMs can be massively beneficial if they are used to aid KGs because understanding and processing human language is a crucial part of KGC. For instance, LLMs can enhance the semantic content of the knowledge graph by comprehending descriptions of entities and their context. They can also be used for link prediction as they can more accurately infer these linkages by utilising the extensive semantic representations they have gained throughout training.

2.5 Performance Evaluation

2.5.1 Knowledge Graph Embedding Models

TransE is a relatively simple KGC technique where it models the relationships between entities as translations in the embedding space (Bordes et al., 2013*b*). All the entities and vectors are represented in a continuous vector space to give a score to the triples. For a triplet $< h, r, t >$ where h and t are the head and tail entities and r is the relationship

between them, the TransE scoring function is defined as:

$$f(h,r,t) = -\|\mathbf{e}_h + \mathbf{r} - \mathbf{e}_t\| \tag{2.4}$$

Here, $\mathbf{e}_h$, $\mathbf{r}$, and $\mathbf{e}_t$ are the embeddings of the head entity, relation, and tail entity, respectively, and $\|\cdot\|$ denotes the L2 norm. The relation vector $\mathbf{r}$ translates the head entity vector $\mathbf{e}_h$ closer to the tail entity vector $\mathbf{e}_t$, capturing the relationship between entities as a geometric translation.

TransE's simplicity to implement and understand makes it a favourite for many people, as it has performed well on various benchmark datasets. It is also computationally efficient, which makes it easier to scale as the size of the KG increases. However, it struggles with complex relationships like one-to-many and many-to-one and is hence unsuitable for more intricate KGs. It makes the assumption that relations can be expressed as translations, which might not fully capture more complex relationships between different entities.

DistMult is another KGC technique which predicts the links in a KG by factorising a 3-dimensional tensor that represents the KG. A diagonal matrix is used to represent relations and calculate a score for each triplet to measure its authenticity (Yang et al., 2014). For a triplet $< h,r,t >$, the scoring function is:

$$f(h,r,t) = \mathbf{e}_h^\top \cdot \mathrm{diag}(\mathbf{r}) \cdot \mathbf{e}_t \tag{2.5}$$

It involves a dot product that captures the interactions between the head entity, relation, and tail entity.

The use of the diagonal matrix makes it straightforward to understand and implement. Similar to TransE, because of its computational efficiency, it is easier to scale according to the size of the graph. It, however, struggles with asymmetric relationships, which directly reduces the model's expressiveness because it does not capture interactions between the embeddings' many dimensions. This might also lead the model to create an oversimplified version of the graph, which can lead to it missing some nuanced links between different entities.

ComplEx is the relatively the most complicated of these methods for KGC. It extends the KG by extending the bilinear model to the complex domain (Trouillon et al., 2016). This is mainly to handle asymmetric relations in the data. It represents the entities and relations as complex-valued vectors and then scores each triplet of $< h,r,t >$. The scoring function is defined as:

$$f(h,r,t) = \mathrm{Re}(\langle \mathbf{e}_h, \mathbf{r}, \overline{\mathbf{e}_t} \rangle) \tag{2.6}$$

where $\mathbf{e}_h$, $\mathbf{r}$, and $\mathbf{e}_t$ are the complex embeddings of the head entity, relation, and tail entity, respectively. $\overline{\mathbf{e}_t}$ is the complex conjugate of $\mathbf{e}_t$, and Re denotes the real part of the resulting complex number. This formulation allows the model to capture both symmetric and asymmetric relations.

While being as simple and easy to implement as the other bilinear models, it extends their expressiveness to the complex segment as well. This enhanced expressiveness, which allows it to model asymmetric relations, helps it set itself apart from other methods. However, adding the complex segments increases its computational requirements, making it more resource-hungry for large KGs. It still might miss higher-order interactions that go beyond pair-wise relations.

RotatE is a KGC technique similar to TransE. It models relations as rotations but in the complex plane. This helps it to handle asymmetric relations as well. This method uses rotational transformations to record the relationships between entities and displays entities and relations as complex-valued vectors (Sun et al., 2019). The scoring function is defined as:

$$f(h,r,t) = -\|\mathbf{e}_h \circ \mathbf{r} - \mathbf{e}_t\| \tag{2.7}$$

Here, $\mathbf{e}_h$, $\mathbf{r}$, and $\mathbf{e}_t$ are the complex embeddings for the head entity, relation, and tail entity, respectively. $\circ$ denotes the element-wise (Hadamard) product, and $\|\cdot\|$ is the L2 norm. The relation $\mathbf{r}$ acts as a rotation in the complex plane, transforming $\mathbf{e}_h$ into a vector that should be close to $\mathbf{e}_t$.

Because of its novel use of rotations in the complex plane, it can model a wide range of relation patterns, including symmetry, antisymmetry, inversion, and composition. Because of this, it is very good at capturing intricate relational structures in knowledge graphs. Similar to ComplEx, the inclusion of the complex segment along with the rotations increases its computational requirements, making it a bit more difficult to scale.

2.5.2 Evaluation Metrics

Mean Reciprocal Rank (MRR) and Hits@k are among the most common evaluation metrics for KGC. These metrics predict the performance of different KGC techniques and provide information on how well a model ranks the right entities or relationships in relation to the wrong ones in its predictions.

The average of the reciprocal ranks of the correct entities or relations across all queries is determined by MRR. The model predicts a ranked list of entities and relation-

ships for every query in the evaluation set [5]. The reciprocal rank for a correct prediction is computed as $\frac{1}{\text{rank}}$, where rank is the position of the correct prediction in the ranked list. MRR is then calculated as follows:

$$\text{MRR} = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{\text{rank}_q} \tag{2.8}$$

where $|Q|$ is the total number of queries, and rank_q is the rank of the correct prediction for query q.

A simple way to look at it is that the higher the MRR value, the higher the chances of correct predictions appearing in a ranked list. This directly measures the accuracy of the model and tells us how well it is performing compared to all queries in the evaluation set.

Hits@k is a relatively simpler way to assess the performance of the models. They give a proportion which tells us the number of queries for which the correct entity or relationship appears within the top-k predictions [6]. For each query, Hits@k counts as a hit if the correct answer appears in the top-k-ranked predictions. The Hits@k metric is defined as:

$$\text{Hits@k} = \frac{1}{|Q|} \sum_{q \in Q} \text{hit}_q, \quad \text{where hit}_q = \begin{cases} 1 & \text{if rank}_q \leq k \\ 0 & \text{otherwise} \end{cases} \tag{2.9}$$

They provide a more in-depth analysis of the model than MRR, as they only consider success within a specified value of k. The value of k is also flexible to suit the needs of the task and depending on the knowledge graph.

[5] https://www.evidentlyai.com/ranking-metrics/mean-reciprocal-rank-mrr

[6] https://typeset.io/questions/how-hits-k-is-used-for-evaluating-missing\-link-prediction-in-o5muhtou40

Chapter 3

Data Preparation

3.1 Understanding the data structure

The first step in any process is preparing the data that is required for all the activities. This scenario is no exception; we first have to process the data of the KG so that the LLMs can read it to provide us with new triples. The KG data is stored in a single `txt` (.text) or `.tsv` (tab-separated values) file. For simplicity, the triples are going to consist of a subject, a predicate, and an object. Here, subject and object are the entities, and the predicate is some relation between them. A KG has multiple entities joined by varying relationships between them. Hence, the same subject can have different relationships with different objects, forming interconnected knowledge.

However, for data and memory efficiency (especially in big datasets), the actual entity name or the relationship name is not used. Instead, they are encoded and represented by some ID. The IDs can be interpreted using a metadata file, which tells us precisely what an ID represents and thus is helpful in interpreting the KG. These IDs are then used to create the embeddings to run KG Embedding models to check KGC performance. We want to boost this performance by enhancing the KG knowledge by adding more triples with the help of LLMs. Figure 3.1 shows an example of the FB15K-237 KG, how it is stored and what the metadata (created from Freebase) for it looks like. In this dataset, the relationship is not encoded; instead, it is divided into different categories.

The challenge now becomes that an LLM will not understand the ID for either the relation or entity without explicitly being told what they stand for. Hence, the need becomes to convert the IDs back to their actual label so that an LLM can form a cohesive understanding of the subject, predicate and object and then accordingly aid in

```
/m/07l450   /film/film/genre    /m/082gq
/m/07h1h5   /sports/pro_athlete/teams./sports/sports_team_roster/team   /m/029q3k
/m/0ydpd    /location/location/time_zones   /m/02hcv8
/m/0738b8   /people/person/places_lived./people/place_lived/location    /m/02xry
/m/070xg    /sports/sports_team/colors  /m/01g5v
/m/09306z   /award/award_ceremony/awards_presented./award/award_honor/honored_for   /m/0sxkh
/m/0kbws    /olympics/olympic_games/participating_countries /m/027jk
/m/0kbws    /olympics/olympic_games/participating_countries /m/04vjh
/m/01c9f2   /award/award_category/winners./award/award_honor/ceremony   /m/01bx35
```

(a) Original Dataset: FB15K-237

```
"/m/010nlt": {
  "alternatives": [],
  "description": "administrative area of the Principality of Monaco",
  "label": "Monte Carlo",
  "wikidata_id": "Q45240",
  "wikipedia": "https://en.wikipedia.org/wiki/Monte_Carlo"
},
```

(b) Metadata for FB15K-237

Figure 3.1: FB15K-237 Dataset Sample

the augmentation of the data. This also becomes challenging because, in datasets like FB15K-237 (Figure 3.1), where some IDs have distinct labels, they can also be known with different labels in other triples. We have to handle these discrepancies before this data can be augmented.

3.2 Process

The process of creating the dataset in a format understandable by an LLM is relatively straightforward. For the scope of this dissertation, we are only going to be working with the FB15K-237, Codex-s, Nations, and UMLS datasets. The Nations and UMLS datasets are relatively small, and all the triples in them are directly in the format of subject, predicate, and object as their own (no IDs); hence, they do not require any preprocessing.

In the case of FB15K-237, we go over each subject and object ID of a particular triple, search the metadata for that ID and replace it with its label. It is to be noted that a fraction of IDs ($\approx 0.007\%$) did not have corresponding data for them in the metadata for FB15K-237. Since we cannot have any IDs in the data before augmentation, these small number of triples were dropped. The IDs in the Codex-s dataset were renamed in a similar manner, but the relationship was also encoded, so we had to replace the

relationship ID with the actual relationship as well.

```
Once Upon a Time in America /film/film/country  Italy
John Lee Hooker /people/person/profession   singer-songwriter
Rango   /film/film/genre    action film
Brad Dourif /people/person/gender   male organism
My Big Fat Greek Wedding    /film/film/genre    romantic comedy
Anjelica Huston /people/person/place_of_birth   Santa Monica
A Beautiful Mind    /film/film/language English
Kaneto Shiozawa /people/person/nationality  Japan
3 Idiots    /film/film/genre    comedy film
```

Figure 3.2: Processed FB15K-237 Dataset

Figure 3.2 shows the post-processed KG data, which can be fed into LLMs. Another helpful feature in the metadata is the description of a particular ID, which can be added to the label to help the LLMs further understand the knowledge represented by the triple. Finally, for entities with more than one label, we ensure that the same label is used across the particular KG. This results in a very minimal loss of accuracy in the KG triples because different forms of the label for a specific ID can add more knowledge in different triples. Now, the data is ready to be augmented by LLMs.

Chapter 4

Generating Augmented Triples

Now, we have the data prepared in a format that the LLMs can interpret so that the models can understand the information in them, which helps us generate meaningful new triples to augment the latest data. This process comes with its own set of challenges. We have to be very precise with the prompt that is being provided to the model so that it understands the task, and this prompt can change depending on the model as well. We also have to ensure that the entities and relations that the model generates are taken directly from the KG itself and the model is not able to create random new components of its own. Keeping these things in mind, we go over two approaches to augment the data available in the KG. Let's look at the two methodologies in detail. For experimentation of the methods, we are going to be using the smaller UMLS and Nations datasets and then implementing the best method on bigger datasets.

4.1 Directly Generate from LLMs

The first and most direct approach is to generate the triples directly from the LLM itself. Since LLMs excel at understanding the intricate details of the data, they'll learn well from the data directly provided to them and generate new triples accordingly. In this method, we will provide the triples available in the KG to the model as input, along with a prompt message to get it to generate triples in the same format and use the same entities and relations already present in the KG.

For running the LLMs, we use the LMStudio application, which is a desktop application primarily used for experimenting with local and open-source LLMs [1]. It is

[1] https://lmstudio.ai/

directly connected to Huggingface [2], which gives it access to all the models available on Huggingface as well. The app provides an interface to chat with an LLM model running locally on a system. It also has the added ability to create a local server with which we can access a particular LLM in our code. The idea here is to send the entire KG as the input along with a prompt to have it generate our new augmented triples based on its understanding of the knowledge already present in the KG.

```
6. National Society of Film Critics Award for Best Supporting Actor - Milk
7. Independent Spirit Award for Best Supporting Female - Winter's Bone
8. The Mummy: Tomb of the Dragon Emperor - Country: People's Republic of China
9. Seiyū - Sailor Moon Super S: The Movie
10. Game Change - Spain
11. Paramount Pictures - The Godfather Part II
12. Golden Globe Award for Best Screenplay - Do the Right Thing
```

Figure 4.1: Summary of the dataset provided instead of triples

However, we immediately encounter a significant challenge with the allowed input tokens for a particular model. Some of the smaller models, like GPT-2 and StableLM, have an input token limit of 1024 tokens. Adding to that, we also have to add the prompt, which will tell the LLM what exactly to do with the input triples and how to handle it. Only adding randomly selected `k` triples would result in us being unable to provide smaller models with the crucial information already available in the KG. Even using Chain-of-thought (CoT) prompting (Wei et al., 2023) feeding the model the triples in batches was not working out great. Overcoming these challenges, the model gives us a summary of the triples instead of generating new triples (Figure 4.1).

```
1. Golden Globe Award for Best Screenplay - Nominated for Do the Right Thing
2. Paramount Pictures - Distributed The Godfather Part II
3. Lauren Holly - Nationality: United States of America (same as Ellen DeGeneres)
4. Rebecca Romijn - Canoodled with Justin Theroux
5. Mickey Rourke - Award nominee for Independent Spirit Award for Best Male Lead
```

Figure 4.2: Unformatted triples directly generated from LLMs

Compromising on the a variety of factors and employing CoT, even if we were able to generate new triples, they were not formatted correctly (Figure 4.2). The newly generated triples have to be appropriately checked to ensure that both the relations and entities already exist in the KG which was not always feasible for big datasets. Since the generation is somewhat uncontrolled by the LLM, the new triples were also adding unwanted information in the KG by the way of new entities and relations.

[2] `https://huggingface.co/`

4.2 Scoring Triples

Since directly generating the triples did not work, we have to look towards another approach. Instead of making the LLMs directly generate new triples for us, we can generate the triples ourselves and ask the model to tell us what the likelihood of the triple actually being actual. The model can use its knowledge to judge and score each new triple. The newly generated triple will be passed to the LLM as input, and the model can tell us the likelihood of its continuation. Scoring each triple here can tell us which triples will be most helpful in augmenting the KG data to improve the KGC score for that particular dataset.

```
A relation_1 B
C relation_2 D
E relation_3 F
G relation_4 H
I relation_5 J
```

(a) Actual Triples

```
E relation_5 H
I relation_1 B
G relation_2 J
A relation_3 D
C relation_4 F
```

(b) New Triples after Shuffling

Figure 4.3: Shuffled Triples Sample

The process will begin by having some artificial triples for the model to score. The simplest way to generate the new artificial triples is by shuffling the subject, predicate and objects of the existing triples, resulting in new triples and verifying they don't already exist in the KG (Figure 4.3). These triples are assured to be made of the same entities and relationships that already exist in the KG, and since they are being shuffled, the format would also not be an issue. These triples will be individually sent as the prompt for the model, where they will be scored and sorted to select only the highest-rated triples as the most likely ones to be added to the KG.

4.2.1 How the scoring works?

Since we are using stock models and not fine-tuning them for the scope of this research, we will be using all models in their inference mode. This ensures that the gradients are not updated, and the model gives the output on the basis of its already trained weights while also making the code more efficient to run. We will be scoring the models

with two open-source models — GPT-2 and Google Gemma 2B [3]. We have used the transformers package to run our LLMs here.

The input of the model is first tokenised so that the model can understand the input. These models have special tokenisers that help them understand the input easily. During inference, the model has to predict the next token in the sequence, given all the previous tokens. In a large language model, it is done so by predicting the probability distribution over the entire vocabulary set for each token position. The model output is a set of logits for each possible token in the vocabulary, indicating how likely that token is given all the tokens that have preceded it in a particular order.

Using these logits, the loss is calculated, which makes the model compare its prediction with the actual tokens in the input sequence. This is the Negative Log-Likelihood (NLL), which is mathematically given as:

$$\text{NLL} = -\sum_{t=1}^{T} \log P(y_t \mid y_{<t}, \text{prompt}) \tag{4.1}$$

Where:

- T is the length of the token sequence.
- y_t is the actual token at position t.
- $P(y_t \mid y_{<t}, \text{prompt})$ is the probability assigned by the model to the token y_t given all preceding tokens and the prompt.

A lower NLL indicates that the model assigns high probabilities to the observed sequence of tokens, suggesting that a particular triple is more likely to be true. The problem with this methodology of directly scoring just the triple is that these scores are entirely reliant on the knowledge of the LLM. Since the model has no information about the information in the model, it is purely using its own knowledge to score the triples passed to it. This can be challenging in domain-specific KGs where the LLM will need to have knowledge of the domain to understand the entities and relationships accurately and then systematically score each triple. Otherwise, the scores can be highly random, and entities or relations that the LLM lacks information about may be scored as equally likely. This will again introduce unwanted information in the KG, hampering the integrity of the information already available.

[3] `https://huggingface.co/google/gemma-2-2b`

4.2.2 In-Context Learning

Simply passing a single triple to the model is not helpful as the scores are very random, and on every iteration, the model was scoring a different set of triples as the most likely ones to be added to a particular KG. This is most likely due to the LLM never having come across input in a similar format or having little knowledge about the domain regarding which the triple is about. LLMs are trained on large amounts of data, but the models will struggle with something they have not seen before (either the domain context of the triple or the format in which they need to be).

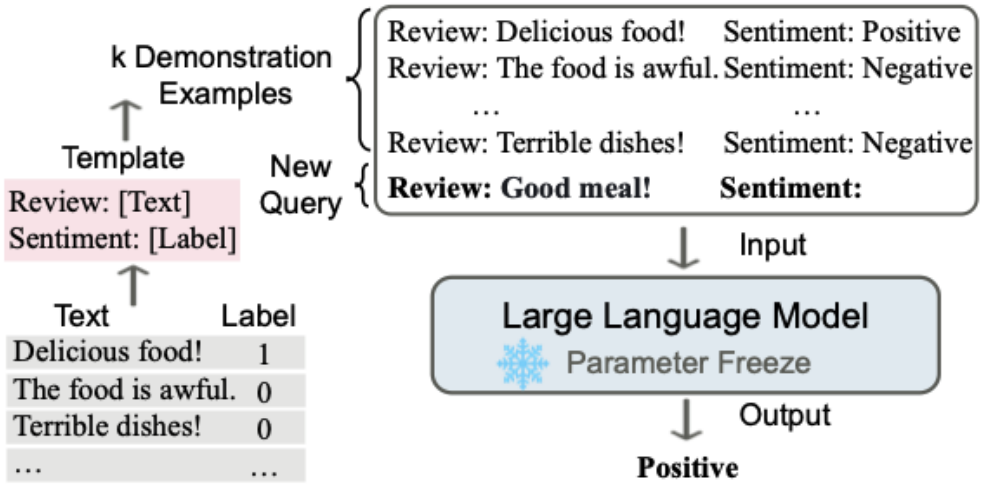


Figure 4.4: In-Context Learning Explained (Dong et al., 2024)

We tackle this issue using In-Context Learning (ICL) (Brown et al., 2020). We pass the LLM a context so that it can properly analyse and learn from it and then do the task that it is asked to do. In this scenario, the input would be the existing triples of the KG. They would provide important context regarding the information in the KG and also tell the KG what the input looks like. Hence, our prompt would become the list of triples available in the group, followed by the randomly generated triple that we have to score. Now, instead of scoring the triple only based on its own information, the LLM can learn from the context provided to it, learn from it and then score the entire continuation for the model.

This increases the model complexity as now, instead of just scoring a single triple, it is scoring the entire context along with one augmented triple to see if the augmented triple is a valid continuation of the already existing triples. Adding the triples to the prompt drastically improved the scoring, and the scoring of the randomly generated triples was less random and relatively more logical. Due to the limitations of the model

token length, the list of triples passed to the model was random but gave the models just enough idea about what information is represented in the KG data and how likely the new triple is. While useful, many unlikely triples were scored highly despite making no sense (Figure 4.5). This was primarily because, for those particular triples, the other triples which were provided as context in the prompt were not related to the new augmented triple, leading to inaccurate scoring.

```
Michel Legrand  diplomatic relation Missile Technology Control Regime
Levon Helm  diplomatic relation International Finance Corporation
Damon Albarn    instrument  United States of America
Jean-Baptiste Dumas languages spoken, written, or signed    United States of America
Greece  occupation  Organisation for Economic Cooperation and Development
Heinrich Brunner    country Arab Fund for Economic and Social Development
```

Figure 4.5: Irrelevant triples being scored highly

To tackle this challenge and improve this methodology further, we decided to precisely select what triples to provide in the context of the model rather than randomly selecting triples from the entire context. This worked as follows — given a new augmented triple to score with subject s, predicate p and object o, only the triples which have either s, p or o in them were selected to be as context triples for that particular new augmented triple. Even in this case, if the input token length issue arose, the randomly chosen triples for context were those that included either s, p or o. This not only made the process more efficient but also provided a much better score for all the randomly generated triples. Hence, the overall pipeline from start to end is shown in Figure 4.6.

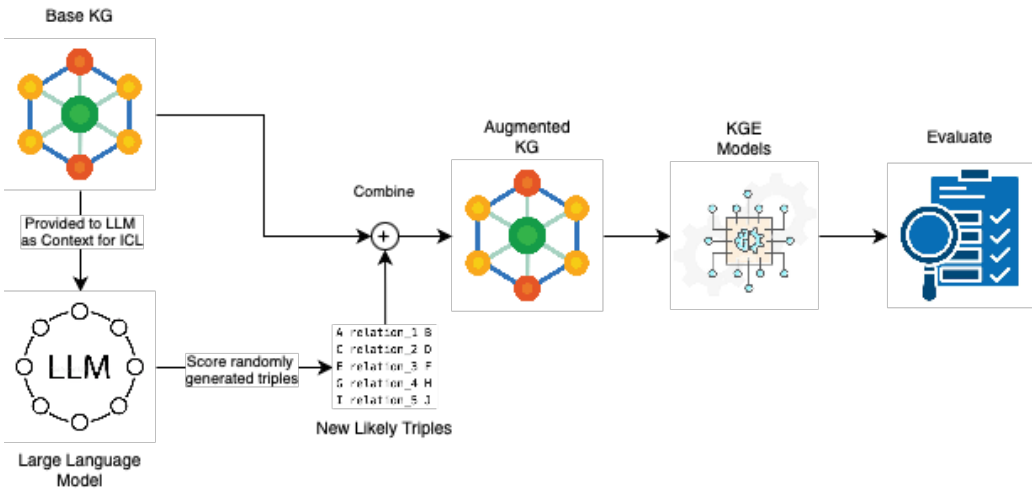


Figure 4.6: Overall experiment pipeline

4.3 Analysis of Generated Triples

Due to the randomness of the generated triples, it is challenging to determine which ones are actually correct and meaningful unless we manually analyse all of them. An interesting observation here is that the model has now begun to score triples which have all s, p and o belonging in the same field highly, even though the triple $< s, p, o >$ might be factually incorrect. This was noted across many fields, especially in the Codex-S and UMLS datasets. Similar to this, triples with subject and object of the same field, but the predicate of some other field were also highly scored, again not necessarily being correct (Figure 4.7).

```
Ernst Curtius   educated at International Centre for Settlement of Investment Disputes
Herbie Hancock  member of   International Bank for Reconstruction and Development
Wolfhart Pannenberg occupation  Organisation for the Prohibition of Chemical Weapons
David Mitchell  member of   Organisation for the Prohibition of Chemical Weapons
Pierre-Joseph Proudhon  member of   International Centre for Settlement of Investment Disputes
Ted Nugent  diplomatic relation Organisation for the Prohibition of Chemical Weapons
Mstislav Keldysh    member of   Multilateral Investment Guarantee Agency
Vince Gill  influenced by   International Centre for Settlement of Investment Disputes
Jive    occupation  International Bank for Reconstruction and Development
```

Figure 4.7: Few somewhat correct triples being scored highly

A comprehensive manual analysis was conducted on approximately 300 selected triples from each dataset incorporated into various collections. Figure 4.8, 4.9, 4.10 and 4.11, presents the findings, highlighting the quality and relevance of the added triples across datasets. The distinct behaviour of the Nations dataset stems from its entire content belonging to a single field.

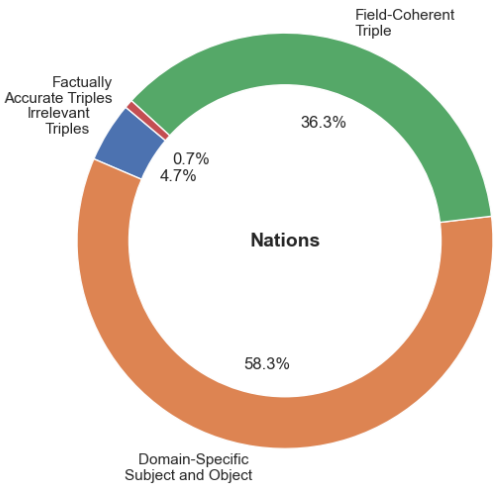


Figure 4.8: Augmented Triple Quality of Nations Dataset

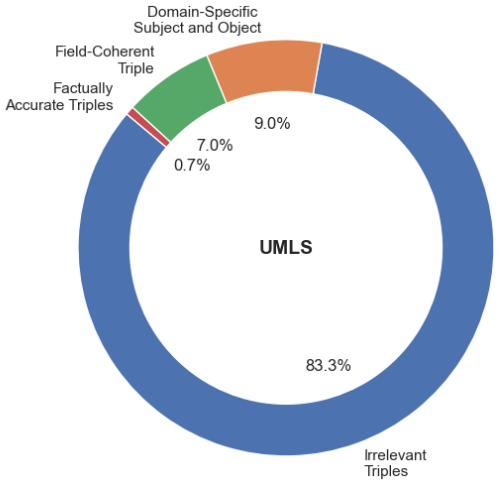


Figure 4.9: Augmented Triple Quality of UMLS Dataset

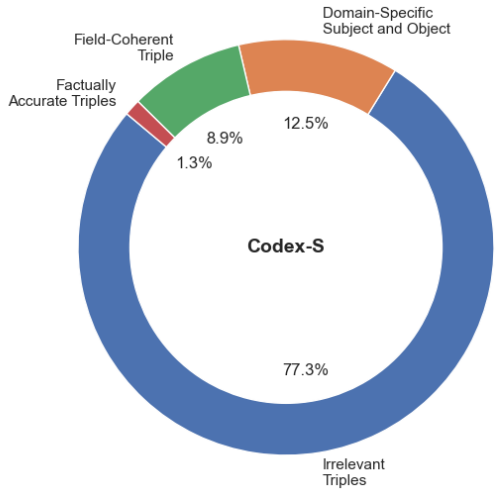


Figure 4.10: Augmented Triple Quality of Codex-S Dataset

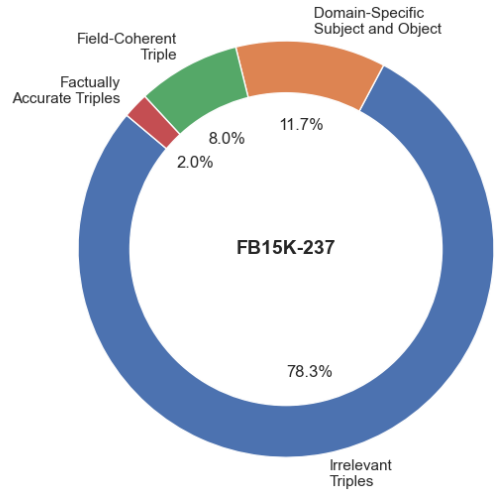


Figure 4.11: Augmented Triple Quality of FB15K-237 Dataset

Chapter 5

Evaluation

We now have the new augmented triples from different LLMs ready. They will be added to the base dataset to see if the addition of the augmented triples increases the performance of KGE models to better predict missing links in the KGC datasets. We will first look at the methodology for adding new links to the existing base datasets and then see what the results of the KGE models look like on the new dataset.

5.1 Methodology

The first stage is to add the augmented triples to the dataset. Again, for the initial testing, we will be using the small UMLS and Nations datasets. We have already generated our triples and have a score for each of them indicating which of them can be most likely to be added to the dataset. Due to the significant disparity in the quality of the triples, we have to be careful when selecting the correct triples to add to the dataset.

We begin by selecting the top 2500 scoring triples for each of the datasets. These definitely include the best human-evaluated triples, and the remaining triples are highly scored from the remaining generated triples. Our goal is to prioritise the correct and partially correct triples from the generated ones because otherwise, we would be adding a lot of incorrect information in the KG. Once these triples are selected, we add them to the base dataset and create a separate dataset which can be fed into the base model. To do thorough testing, we checked by adding a different number of augmented triples to the dataset.

We initially started by adding 500 and 2500 augmented triples to both the base datasets. We only used the triples scored by the GPT-2 model for initial testing. Now, it is time to run the KGE models to see some actual results. The code was also ran on the

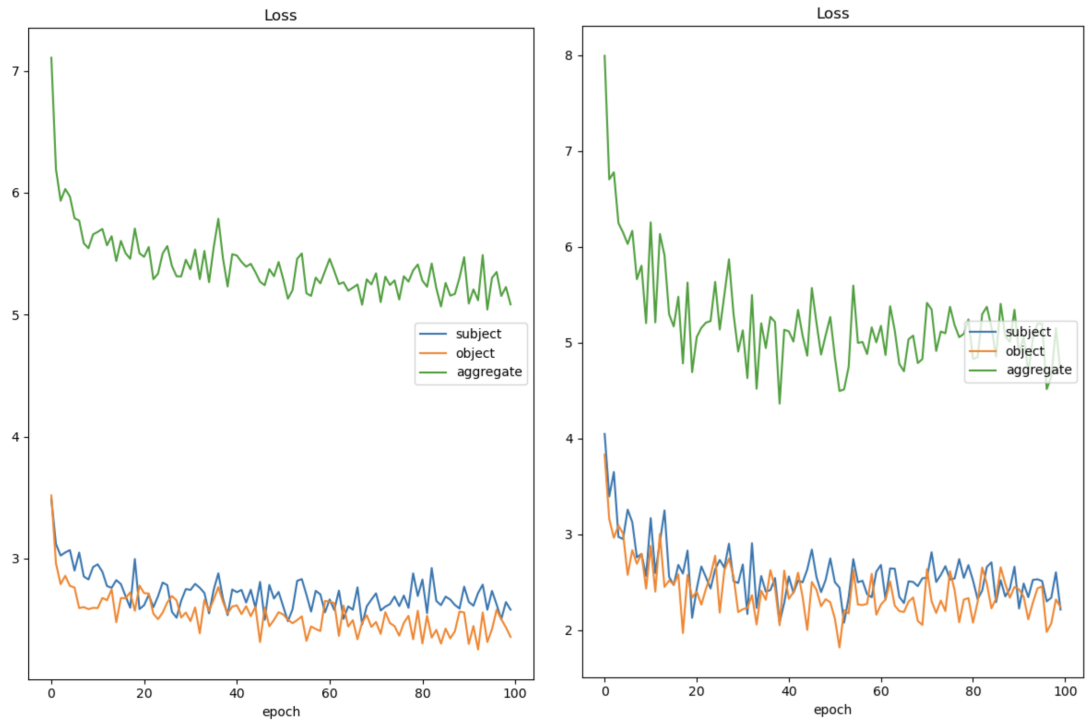
base dataset to compare the with the performance on the augmented dataset. We picked 2 KGE models of DistMult (Yang et al., 2014) and ComplEx (Trouillon et al., 2016) due to their different methods of predicting links in the KG. Our main point of comparison will be the Mean Reciprocal Rank (MRR) of the model. The embedding size of the model was kept at 500 tokens, and it was trained for 100 epochs with a batch size of 128. The model used the Adam Optimiser (Kingma and Ba, 2017) with a learning rate set at 0.1. The results of the experiments are listed in Table 5.1.

Triples Added	**UMLS**		**Nations**	
	DistMult	**ComplEx**	**DistMult**	**ComplEx**
Base	**70.4**	**95.7**	**76.1**	**79.6**
500	66.1	93.1	67.8	74.5
2500	55.0	87.5	72.3	76.7

Table 5.1: Initial KGC Experiments

We see some interesting results in our initial test. Firstly, ComplEx, although being slightly more computationally expensive, outperforms DistMult in all the benchmarking. This was the indication that only the ComplEx model should be used for all further testing. As we can see, the base dataset outperformed the augmented dataset with both 500 and 2500 added triples. After a quick analysis, this can primarily be attributed to the small size of both datasets. The UMLS dataset has just 5,216 triples in the training set, and the training set for the Nations dataset has just 1,591 triples. Hence, adding 2,500 or even 500 triples means we are adding close to 33% of the training size for Nations and 9.5% of the training size for the UMLS dataset back into the data.

This data need not be accurate and hence adds a lot of redundant and even wrong information to the KG, polluting the information already available in it. This can be further validated by the loss comparison of the UMLS dataset while training, in which we can see the loss is declining in a relatively smoother manner as compared to the added dataset (Figure 5.1). We need to adjust the added number of triples in accordance with the dataset size to ensure this does not happen. Hence, now we add a small number of triples to the smaller datasets and gradually increase them to see how the addition of more triples affects the model performance. Another benefit of this method is that it can help us ensure that the quality of small triples that are added to the data are all of the highest quality. Further hyperparameter tuning is also performed to ensure that the best set of parameters are selected for the testing of the methodology.



(a) Base UMLS Dataset (b) UMLS Dataset with 2,500 Augmented Triples

Figure 5.1: Loss during KGE model training on UMLS Dataset

5.2 Results

Now that we have a clear idea about the approach to take regarding the number of triples to add to the base dataset and the hyperparameters for running the models, we start experimenting on all the datasets while keeping their size in mind. We use 3 smaller datasets — Nations, UMLS and Codex-s and one extensive dataset — FB15K-237.

For the smaller datasets, we gradually increase the number of added triples in each dataset to see how it affects the KGC performance. We add 10, 25, 50, 100, 150 and finally 200 triples to them. For the bigger dataset, we add 500, 1500, and 2500 triples because adding a small number of triples would barely make a difference to its dataset, and we would not see any meaningful outcomes. In a similar manner, we kept the batch size for the smaller datasets at 128 while training but increased it to 4192 for the bigger models to ensure training at a decent pace. The embedding size remained unchanged at 500 with a learning rate of 0.1 using the Adam optimiser. During initial testing, we saw that the model performance was not improving a certain number of epochs. Hence, we reduced the number of epochs for small datasets to 50 and 10 for bigger datasets. A summary of the experimental setup is shown in Table 5.2.

Dataset	Size Category	Number of Triples Added	Batch Size	Epochs
Nations	Small	10, 25, 50, 100, 150, 200	128	50
UMLS	Small	10, 25, 50, 100, 150, 200	128	50
Codex-s	Small	10, 25, 50, 100, 150, 200	128	50
FB15K-237	Large	500, 1500, 2500	4192	10

Table 5.2: Summary of Experimental Setup for Datasets

We only checked the performance using the ComplEx KGE model due to its superiority in handling edge cases better and, despite being a bit more resource-heavy, significantly faster than the DistMult model. Finally, using the experimental setup we described, we generated triples from both GPT-2 and the Google Gemma 2B model. The GPT-2 model has approximately 117 million parameters, whereas the Gemma has close to 2.5 billion parameters. This will give us an idea regarding how the model size affects the scoring of the triples and if it is any better to use a bigger model or if a smaller model will work just fine for this use case. All the datasets are evaluated using their MRR score on the validation and the test set. All the results for both the models are shown in Table 5.3 and 5.4 for small datasets and in Table 5.5 for the large dataset. Once we are sure of the added number of triples and the model which gives the best results, we will use that on The Dock data to see if the results we see here are translated to an actual drug interaction KG.

Triples Added	Nations		UMLS		Codex-s	
	Val	Test	Val	Test	Val	Test
Base	0.8300	0.8200	0.9660	**0.9650**	0.4658	0.4516
10	**0.8377**	0.8137	0.9618	0.9603	0.4696	0.4583
25	0.8121	0.8204	**0.9664**	0.9648	0.4721	0.4601
50	0.8134	**0.8242**	0.9606	0.9631	0.4728	0.4589
100	0.7977	0.8111	0.9548	0.9544	0.4720	**0.4624**
150	0.8007	0.8176	0.9576	0.9633	0.4738	0.4605
200	0.7911	0.7989	0.9591	0.9608	**0.4775**	0.4606

Table 5.3: All MRRs after augmenting datasets from triples scored by GPT-2 Model

Triples Added	Nations		UMLS		Codex-s	
	Val	**Test**	**Val**	**Test**	**Val**	**Test**
Base	**0.83**	**0.82**	0.966	0.965	0.4658	0.4516
10	0.8233	0.8118	0.9658	0.9641	**0.4758**	**0.4633**
25	0.8079	0.8177	0.9601	0.9616	0.4712	0.4578
50	0.8138	0.817	0.961	0.9608	0.4747	0.4602
100	0.7964	0.8041	**0.9665**	**0.966**	0.4744	0.4618
150	0.8199	0.8069	0.9661	0.9658	0.4685	0.4529
200	0.7886	0.784	0.964	0.9607	0.4744	0.4585

Table 5.4: All MRRs after augmenting datasets from triples scored by Gemma Model

Triples Added	GPT-2		Gemma	
	Val	**Test**	**Val**	**Test**
Base	0.3204	0.3146	**0.3204**	**0.3146**
500	0.3211	0.3165	0.3199	0.3127
1500	**0.3242**	**0.3175**	0.3176	0.3138
2500	0.3076	0.3018	0.3044	0.3000

Table 5.5: All MRRs for FB15K-237 dataset across different models

We can see straightaway that augmenting the dataset has helped the performance of the KGE models in some scenarios. We are only looking at the MRR of all the models for this evaluation. An initial observation with the smaller datasets is that the model is performing better with a lower number of added triples in the dataset. Nations and UMLS datasets are performing better, with only up to 50 triples added to the existing data. This approach adds between 1% and 3% more triples to the dataset. Adding a small number of triples also increases the chances of the triples being added to the dataset to be correct or more relevant than triples, compared to ones which do not make sense at all. Despite this modest increase, we observe a marked improvement in performance. However, there is a risk of introducing redundant data into the Knowledge Graph (KG), which can negatively impact the scoring. This trend becomes evident when we add more than 50 triples as performance begins to decline for these smaller datasets.

If we focus on a single dataset specifically, we can compare the performance of the different LLMs which scored the triples. This will help us gauge how different models score triples differently and how their scored triples ultimately affect the value of the augmented triple. From Tables 5.3, 5.4 and 5.5 we can see that the triples scored by the GPT-2 model were enhancing the the datasets comparatively better than the triples scored by the Gemma model. In a typical experimental setup, this observation can be attributed to the random generation of different triples, with each triple being scored based on its subject, predicate, and object and whether they made sense within the given context. To take the randomness out of the equation, we used the same randomly generated triples, scored them using the separate models and then used the top-scoring triples to augment the dataset for the Nations dataset. The KGE models were then run on these datasets, and the comparative performance of the different LLMs with identical experimentation setup is shown in Figure 5.2.

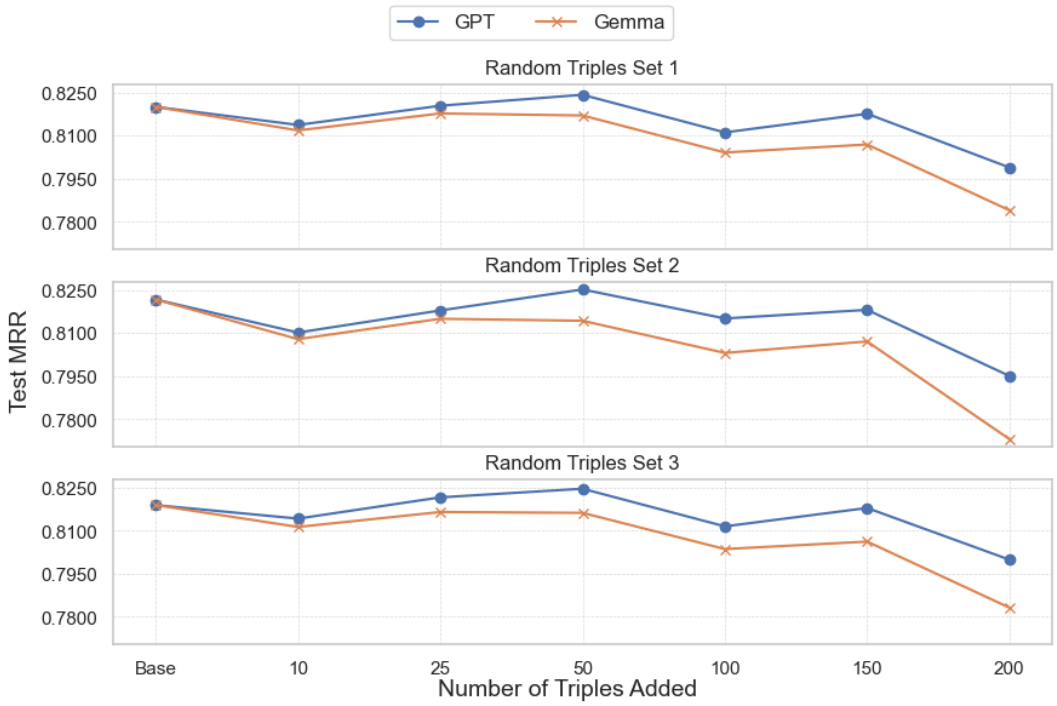


Figure 5.2: Different sets of same random triples scored by different models on Nations dataset

Our earlier observations are solidified by these findings, which again show that the GPT-2 model outperforms the Google Gemma 2B model. It also sticks with the pattern noticed earlier, where adding only 50 triples helps the dataset, and then the performance starts to dip. Similar performance was seen by taking different sets of random triples

with the GPT-2 triumphing over the Gemma model (Figure 5.2). The performance is similar across the runs as the quantity of added triples is low and this ensures that mostly highly scored relevant triples are added to the dataset. This performance disparity, while significant, is not a sign of worry that a bigger model is underperforming compared to a smaller model. It can be attributed to the bigger picture for which these models have been designed. The GPT-2 model might excel at certain tasks compared to the Gemma model, which has been designed for tasks that are much more complicated than this. Fine-tuning of both these models can help us better score these triples and make both the models' performance comparable.

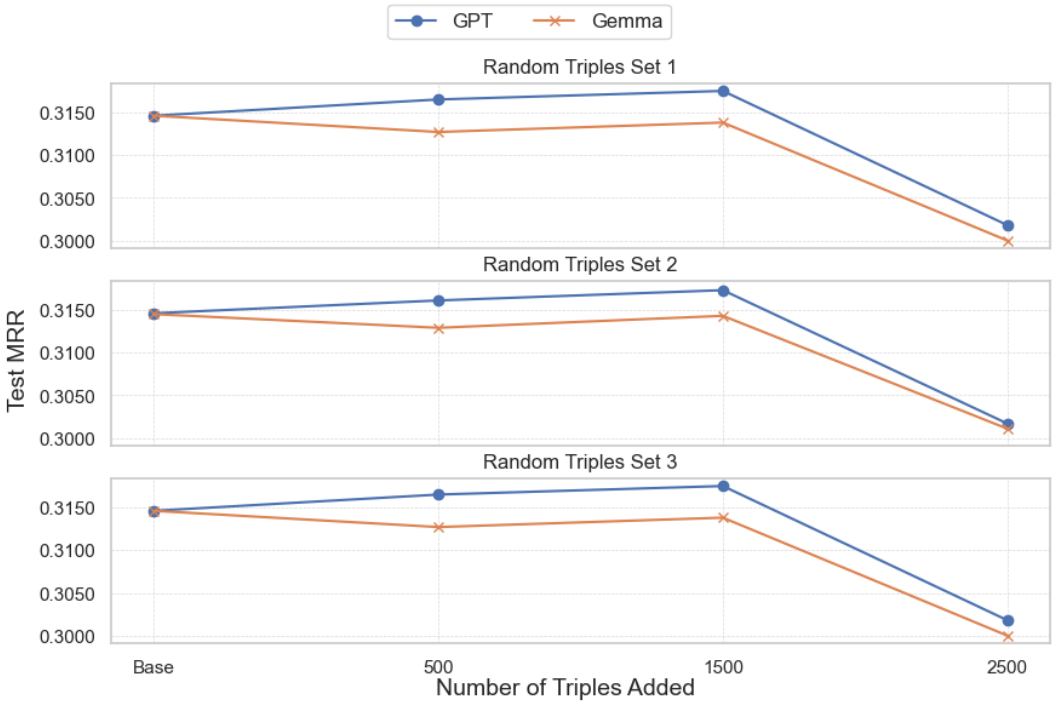


Figure 5.3: Different sets of same random triples scored by different models on FB15K-237 dataset

Heading to the bigger dataset FB15K-237, we notice similar performance as other datasets even though the added triples in this scenario are a lot more. The added triples in this dataset are a lot more to adjust for the large number of training triples in the dataset. Even by adding 0.4% more triples in the training set, we see marginal improvements in the model performance. But as before, adding too many triples and the model performance starts to decline again due to less high-quality meaningful triples and more redundant triples being added to the dataset. The performance of the different runs is very similar for this dataset because the number of added triples is very

minimal considering the size of the dataset. Here again, the performance of the GPT-2 scored triples is better than the Gemma scored triples. This was corroborated by again using different sets of the same random triples and scoring them using the two models to see how it affects the performance (Figure 5.3). In addition to helping us see the performance difference between the models, it strengthens our point that the addition of augmented triples is able to enhance the KGC performance on the dataset further.

5.3 The Dock (Accenture) Data - Drug Interaction KG

We have proved that augmenting the dataset with artificial triples helps improve the performance of KGC. We will now implement this methodology on an actual drug interaction KG provided by The Dock. The data includes triples where a particular drug has relation to a condition. It is a large dataset with 108,482 triples in the train set and 10,332 triples each in the validation and test set. It consists of 7,105 different drugs and 4,682 conditions. The conditions are of two different types: a disorder or a morphological abnormality. Unlike the other KGs, there is no possibility that a subject of one triple is the object of another triple. It is strictly in the form that a subject is a drug, and the object is a condition. There is no information in the predicate of the triple either, as all the predicates are "`has relation to`". This indicates that a particular drug is connected to a condition without giving details as to what sort of condition it is. A sample of the KG is shown in Figure 5.4.

```
CHEK2   has relation to Neoplasm of colon (disorder)
MAPK10  has relation to Familial idiopathic pulmonary fibrosis (disorder)
RPL5    has relation to Reticulosarcoma morphology (morphologic abnormality)
PAXIP1  has relation to De Lange syndrome (disorder)
RHOA    has relation to Genetic disease (disorder)
PSMC1   has relation to Malignant tumor of esophagus (disorder)
CACNG1  has relation to Anxiety neurosis (finding)
```

Figure 5.4: Drug Interaction dataset sample

We will try to improve the KGC score on this dataset by using the same technique as earlier. We will again create random triples and score them using an LLM to see which triples will help us add more value to the information already available in the KG. Since this is not a benchmarking dataset, we will use both the GPT-2 and Gemma models to score the triples to see if they are able to enhance the dataset. Due to the size of the dataset being reasonably large, we will use the same hyperparameters used for the FB15K-237 dataset. We keep the embedding size at 500, utilising the Adam

Optimiser, with the learning rate set at 0.1. The batch size was kept at 4096 with the KGE model training for 10 epochs (we saw a plateau after training for just 6-7 epochs for the FB15K-237 dataset). We again added 500, 1500 and 2500 triples to see how adding more triples affects the performance. All the results are shown in Table 5.6.

Triples Added	**GPT-2**		**Gemma**	
	Val	**Test**	**Val**	**Test**
Base	0.6900	0.7070	0.6900	0.7070
500	0.6823	0.7043	0.6822	0.7023
1500	0.6754	0.6954	0.6706	0.6932
2500	0.6373	0.6521	0.6340	0.6513

Table 5.6: All MRRs for Drug Interaction KG across different models

We can clearly see that adding triples is hampering the score of KGC even with 500 triples, which is just 0.4% of the train set. The performance continues to decline as more triples are further added to it. The main reason for this can be attributed to a lack of information on the relationship between the subject and the object. Since all the relations between the subject and the object are the same, the LLM, while scoring random triples, is not fully able to understand what the triple represents. Hence the LLM

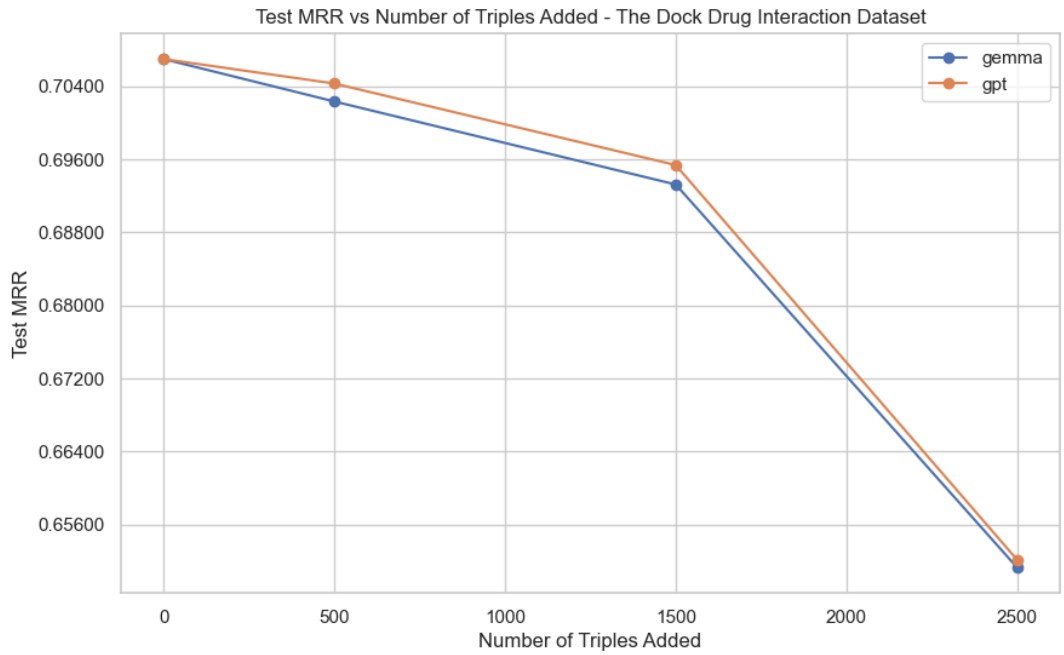


Figure 5.5: All MRRs for Drug Interaction dataset across different models

is left to simply learn from the data by matching which subject (drug) belongs to which object (condition). Now, when an artificial triple comes for scoring with the context including all the triples with the same subject and object, if there is no match, the triple is scored very low. If there was some understanding of the kind of relation between the subject and object, the triple could be scored on the basis of some complex relation via multiple subjects and objects. Hence, most of the triples are low-scored, which means that most triples make little to no sense and, rather than helping the score, impact it in a negative way. The comparison of test MRR of both the models after scoring the same set of random triples again indicates that GPT-2 scored triples outperform the triples scored by the Gemma model by a small model (Figure 5.5). This highlights that both the models are struggling with one single predicate (relation) and no more information about it.

Chapter 6

Conclusions

In this dissertation, we have discussed how using LLMs can enhance the performance of the neural link predictors (KGE models) by augmenting the triples in the KG with the help of LLMs. Initially, we tried to generate the augmented triples from LLMs directly, but due to several complications, we resorted to generating random triples manually and instead used the LLMs to score the triples by providing them sufficient context. This approach resulted in modest improvements in KGE performance as additional triples were added, but beyond a certain point, the performance began to decline as too many triples were introduced. Applying this methodology to an actual KG, the results were less effective, primarily due to missing relational information within the KG.

6.1 Limitations

The main limitation of this methodology is the randomness of the triples generated. Since the pipeline simply shuffles to create random triples without any relevance or context check, there is a good chance that in some or worst-case scenarios, all the triples generated would be completely irrelevant. The pipeline will still take the bad triples and add them to the dataset.

Another huge limitation is the fact that LLMs are still considered black-box [1] models, and there is no clear explanation or transparency regarding how they score the triples. This opacity sometimes makes it challenging to trust the score assigned to the triples, especially in domain-specific contexts where precise and accurate knowledge is crucial.

Another significant barrier is the significant processing time and computer power

[1] https://www.investopedia.com/terms/b/blackbox.asp

needed to execute big LLMs. Due to these models' substantial resource requirements, scoring a high number of triples for larger KGs is both expensive and time-consuming.

6.2 Future Scope

In the future, the generation of triples can be more specific by grouping triples in particular domains and then shuffling them to create random triples. This will likely lead to higher relevant triples, which will increase the quality of the augmented dataset. Using domain-specific knowledge can help us ensure that the generated triples have a higher chance of fitting in with the existing structure and content of the KG.

Exploring other LLMs is another viable option to see if they improve the scoring of the triples and get us better results. Different LLMs have different strengths, and we can find one that can score the triples much better with the context it is provided. We can also try to fine-tune an LLM model using the specific KG we want to improve performance on. Fine-tuning an LLM using data that contains information about the KG can help the model familiarise itself with crucial contextual information that it can use to score triples.

Finally, we can also focus further focus on optimising the hyperparameters for each part of the pipeline. We can directly influence the models' ability to process and interpret triples, leading to more accurate scoring and better-augmented datasets. This could be done by using additional different learning rates and regularisation techniques to optimise the KGE models' performance.

Bibliography

Adamic, L. A. and Adar, E. (2003), 'Friends and neighbors on the web', *Social networks* **25**(3), 211–230.

Arakelyan, E., Daza, D., Minervini, P. and Cochez, M. (2021), Complex query answering with neural link predictors, *in* 'International Conference on Learning Representations'.
URL: *https://openreview.net/forum?id=Mos9F9kDwkz*

Bates, D. W., Cullen, D. J., Laird, N., Petersen, L. A., Small, S. D., Servi, D., Laffel, G., Sweitzer, B. J., Shea, B. F. and Hallisey, R. (1995), 'Incidence of adverse drug events and potential adverse drug events: implications for prevention', *JAMA* **274**(1), 29–34.

Bordes, A., Glorot, X., Weston, J. and Bengio, Y. (2014), 'A semantic matching energy function for learning with multi-relational data: Application to word-sense disambiguation', *Machine learning* **94**, 233–259.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J. and Yakhnenko, O. (2013*a*), 'Translating embeddings for modeling multi-relational data', *Advances in neural information processing systems* **26**.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J. and Yakhnenko, O. (2013*b*), Translating embeddings for modeling multi-relational data, *in* 'Advances in Neural Information Processing Systems', Vol. 26, pp. 2787–2795.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I. and Amodei, D. (2020), 'Language models are few-shot learners'.
URL: *https://arxiv.org/abs/2005.14165*

Destandau, M. and Fekete, J.-D. (2021), 'The missing path: Analysing incompleteness in knowledge graphs', *Information visualization* **20**(1), 66–82.

Dettmers, T., Minervini, P., Stenetorp, P. and Riedel, S. (2018), Convolutional 2d knowledge graph embeddings, *in* 'Proceedings of the AAAI conference on artificial intelligence', Vol. 32.

Dong, Q., Li, L., Dai, D., Zheng, C., Ma, J., Li, R., Xia, H., Xu, J., Wu, Z., Chang, B., Sun, X., Li, L. and Sui, Z. (2024), 'A survey on in-context learning'.
URL: *https://arxiv.org/abs/2301.00234*

Ehrlinger, L. and Wöß, W. (2016), 'Towards a definition of knowledge graphs.', *SEMANTiCS (Posters, Demos, SuCCESS)* **48**(1-4), 2.

Espinosa-Bosch, M., Santos-Ramos, B., Gil-Navarro, M. V., Santos-Rubio, M. D., Marín-Gil, R. and Villacorta-Linaza, P. (2012), 'Prevalence of drug interactions in hospital healthcare', *Int. J. Clin. Pharm.* **34**(6), 807–817.

Farrugia, L., Azzopardi, L. M., Debattista, J. and Abela, C. (2023), 'Predicting drug-drug interactions using knowledge graphs', *arXiv.org* .

Hamilton, W. L., Ying, R. and Leskovec, J. (2018), 'Inductive representation learning on large graphs'.
URL: *https://arxiv.org/abs/1706.02216*

Hripcsak, G. and Albers, D. J. (2011), 'Next-generation phenotyping of electronic health records', *Journal of the American Medical Informatics Association* **20**(1), 117–121.

Jaccard, P. (1901), 'étude comparative de la distribution florale dans une portion des alpes et des jura', *Bulletin de la Société Vaudoise des Sciences Naturelles* **37**, 547–579.

Kingma, D. P. and Ba, J. (2017), 'Adam: A method for stochastic optimization', *arXiv.org* .

Lacroix, T., Usunier, N. and Obozinski, G. (2018), Canonical tensor decomposition for knowledge base completion, *in* 'International Conference on Machine Learning', PMLR, pp. 2863–2872.

Lee, S. Y., Bu, F., Yoo, J. and Shin, K. (2023), 'Towards deep attention in graph neural networks: Problems and remedies'.
URL: *https://arxiv.org/abs/2306.02376*

Naveed, H., Khan, A. U., Qiu, S., Saqib, M., Anwar, S., Usman, M., Akhtar, N., Barnes, N. and Mian, A. (2024), 'A comprehensive overview of large language models', *arXiv.org* .

Newman, M. E. J. (2003), 'The structure and function of complex networks', *SIAM Review* **45**(2), 167–256.
URL: *http://dx.doi.org/10.1137/S003614450342480*

Nickel, M., Murphy, K., Tresp, V. and Gabrilovich, E. (2015), 'A review of relational machine learning for knowledge graphs', *Proceedings of the IEEE* **104**(1), 11–33.

Radford, A. and Narasimhan, K. (2018), Improving language understanding by generative pre-training.
URL: *https://api.semanticscholar.org/CorpusID:49313245*

Safavi, T. and Koutra, D. (2020), CoDEx: A Comprehensive Knowledge Graph Completion Benchmark, *in* B. Webber, T. Cohn, Y. He and Y. Liu, eds, 'Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)', Association for Computational Linguistics, Online, pp. 8328–8350.
URL: *https://aclanthology.org/2020.emnlp-main.669*

Sheth, A., Padhee, S. and Gyrard, A. (2019), 'Knowledge graphs and knowledge networks: the story in brief', *IEEE Internet Computing* **23**(4), 67–75.

Singhal, A. (2012), 'Introducing the knowledge graph: Things, not strings, google', Available at `https://blog.google/products/search/introducing-knowledge-graph-things-not/`. Accessed: 12 April 2024.

Sun, Z., Deng, Z.-H., Nie, J.-Y. and Tang, J. (2019), Rotate: Knowledge graph embedding by relational rotation in complex space, *in* 'International Conference on Learning Representations (ICLR)'.

Toutanova, K. and Chen, D. (2015), Observed versus latent features for knowledge base and text inference, *in* 'Proceedings of the 3rd workshop on continuous vector space models and their compositionality', pp. 57–66.

Trouillon, T., Welbl, J., Riedel, S., Gaussier, É. and Bouchard, G. (2016), Complex embeddings for simple link prediction, *in* 'International conference on machine learning', PMLR, pp. 2071–2080.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. and Polosukhin, I. (2017), 'Attention is all you need', *Advances in neural information processing systems* **30**.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q. and Zhou, D. (2023), 'Chain-of-thought prompting elicits reasoning in large language models'.
URL: *https://arxiv.org/abs/2201.11903*

Wei, Y., Wang, X., Nie, L., Li, S., Wang, D. and Chua, T.-S. (2022), 'Causal inference for knowledge graph based recommendation', *IEEE Transactions on Knowledge and Data Engineering* .

Xu, K., Hu, W., Leskovec, J. and Jegelka, S. (2019), 'How powerful are graph neural networks?'.
URL: *https://arxiv.org/abs/1810.00826*

Yang, B., Yih, W.-t., He, X., Gao, J. and Deng, L. (2014), 'Embedding entities and relations for learning and inference in knowledge bases', *arXiv preprint arXiv:1412.6575* .

Zhang, S., Tong, H., Xu, J. and Maciejewski, R. (2019), 'Graph convolutional networks: a comprehensive review', *Computational Social Networks* **6**(1), 1–23.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C. and Sun, M. (2020), 'Graph neural networks: A review of methods and applications', *AI open* **1**, 57–81.