Comparison of Knowledge Extraction from Knowledge Graphs and DREAM Model for Enhancing Commonsense QA

Hossein Kargar
hkargar17@gmail.com

Ali Manavi
alimanavi22@gmail.com

Abstract

The challenge of enhancing commonsense reasoning and figurative language understanding in natural language processing (NLP) models remains a significant research problem. This project aims to address this issue by comparing two methodologies: the Knowledge Graph (KG) approach and the DREAM-FLUTE method. The Knowledge Graph approach leverages Dense Passage Retrieval (DPR) and similarity comparisons with explanation embeddings, while the DREAM-FLUTE method integrates the DREAM model with T5-BASE and BERT models to improve performance on the Commonsense QA task. Experimental results show that both methods enhance model performance, but they do so in complementary ways, highlighting the potential for integrating these approaches to advance NLP models' capabilities.

1 Introduction

Injecting knowledge into Large Language Models (LLMs) is a critical task to enhance their understanding and reasoning capabilities. This report compares two distinct approaches for knowledge injection: using a Knowledge Graph (KG) and the DREAM model. The Knowledge Graph approach leverages Dense Passage Retrieval (DPR)[Kar+20] and similarity comparisons with explanation embeddings, while the DREAM model integrates elaborations into models like T5 and BERT to improve commonsense reasoning and figurative language understanding.

2 Related Works

2.1 Graph-Based Question Answering

Graph-based methods have been widely explored in question answering (QA) systems. These methods represent knowledge as graphs, where nodes correspond to entities and edges represent relationships between entities. Graph-based QA models leverage these representations to perform reasoning over structured knowledge bases, enabling them to answer complex questions that require understanding of relationships between entities. Recent advancements in graph neural networks (GNNs) have further improved the effectiveness of graph-based QA systems by enabling them to capture higher-order dependencies and perform reasoning over large-scale knowledge graphs[Yih+15].

2.2 Triplet Retrieval for QA

Triplet retrieval methods have been proposed as a means to enhance QA systems by augmenting textual data with structured knowledge. These methods extract triplets (subject, relation, object) from textual data and use them to construct knowledge graphs, which serve as a basis for reasoning. By integrating triplet retrieval with QA models, researchers aim to bridge the gap between textual data and structured knowledge representations, enabling QA systems to perform more effective reasoning over complex queries.[ZZW19]

2.3 Large Language Models (LLMs) for QA

Large Language Models (LLMs), such as BERT, GPT, and T5, have demonstrated remarkable performance in various natural language processing tasks, including QA. These models are pre-trained on large-scale text corpora and fine-tuned on QA datasets to perform question answering. LLMs leverage their understanding of language semantics and context to generate accurate and relevant answers to user queries. Recent research has focused on enhancing LLMs' QA capabilities through techniques such as prompt engineering, knowledge integration, and few-shot learning.

2.4 Knowledge-Augmented QA

Knowledge-augmented QA approaches integrate external knowledge sources, such as knowledge graphs, ontologies, or domain-specific databases, into QA systems to enhance their performance. These approaches leverage structured knowledge to supplement textual data and improve the accuracy and coverage of QA systems. By combining textual and structured knowledge representations, knowledge-augmented QA models can effectively address a wide range of questions that require access to external knowledge.[Mih+18]

3 Approach

3.1 Knowledge Graph Approach

3.1.1 Methodology

The Knowledge Graph approach involves the following steps:

1. Concatenate Query and Choices: For each question, concatenate it with each possible choice to form query-choice pairs.

2. Retrieve Sentences Using DPR: Use DPR to retrieve top 10 sentences for each query-choice pair from a corpus.

3. Compare Similarity with Explanation Embeddings: Compare the retrieved sentences with explanation embeddings.

4. Select Best Query and Choice Pair: Identify the pair with the highest similarity score.

5. Passage Reranking: Use a pre-trained cross-encoder to rerank the sentences.

6. Model Processing with T5 and BERT: Provide the query, choice, and top-ranked sentences to advanced models for final processing.

3.1.2 Detailed Process

1. Concatenate Query and Choices: Combine the question with each choice to create contextually enriched queries.

2. Retrieve Sentences: DPR retrieves relevant passages based on dense vector similarities.

3. Similarity Comparison: Measure similarity between retrieved sentences and explanation embeddings.

4. Best Pair Selection: Choose the query-choice pair with the highest similarity score.

5. Reranking: Cross-encoder reranks the sentences based on relevance.

6. Final Processing: Advanced models like T5 and BERT generate or validate the final answer.

3.1.3 Example

Question: There are 10 apples on an apple tree. Three fall off. Now there are X apples. What is this an example of?
Choices: A) park, B) coloring book, C) garden center, D) math problem, E) gravity
Concatenated Pairs: Each query-choice combination is processed to retrieve and rank the most relevant passages.

3.2 DREAM Model Approach

3.2.1 Overview

The DREAM model enhances commonsense reasoning by generating detailed elaborations for questions, which are then integrated into models like T5 and BERT.

3.2.2 Methodology

1. DREAM-FLUTE System 1: Fine-tunes BERT on the COS-E dataset to improve baseline performance.

2. DREAM-FLUTE System 2: Uses the DREAM model to generate elaborations that are fed into T5-BASE for enhanced predictions.

3.2.3 Detailed Process

1. BERT Fine-Tuning: BERT is fine-tuned on the COS-E dataset to predict correct answers from multiple choices.

2. Elaboration Generation: The DREAM model generates detailed context elaborations.

3. Integration with T5: These elaborations are combined with the original context, question, and choices to improve T5's predictive accuracy.

3.2.4 Example

COS-E Dataset: Provides scenarios, questions, choices, and explanations to train and evaluate models.
Elaborations: Generated by DREAM to provide additional context and improve answer accuracy.

4 Dataset

4.1 Description

Dataset: ConceptNet Knowledge Graph[SCH17]

Source: ConceptNet aggregates data from multiple sources, including WordNet, Wiktionary, OpenCyc, and crowd-sourced inputs from the Open Mind Common Sense project.

Statistics:

- **Size:** ConceptNet 5.7 contains approximately 34 million edges.
- **Languages:** Over 50 languages are supported, with English being the primary language.
- **Nodes and Edges:** The dataset includes a diverse set of nodes representing concepts (words/phrases) and edges representing relationships between them.

Task Description: The task involves using the ConceptNet knowledge graph for question answering (QA) through triplet retrieval. The objective is to extract relevant triplets from ConceptNet that help in answering a given question. The challenges include matching the question to the most relevant triplets and utilizing these triplets to generate accurate answers.

Challenges:

- **Ambiguity:** Questions and concepts may have multiple meanings or interpretations.
- **Noise:** ConceptNet may contain noisy or less relevant edges, complicating the retrieval of useful triplets.
- **Complexity:** Understanding complex relationships and reasoning over multiple edges to generate an answer.
- **Language Variability:** The dataset includes multiple languages and variations in expression, adding to the complexity.

Examples:

- **Input Question:** *What do people use a knife for?*
  - **Relevant Triplet:**
    - Subject: */c/en/knife*
    - Relation: *UsedFor*
    - Object: */c/en/cutting*
  - **Output Answer:** *A knife is used for cutting.*
- **Input Question:** *What is a part of a car?*
  - **Relevant Triplet:**
    - Subject: */c/en/car*
    - Relation: *PartOf*
    - Object: */c/en/engine*
  - **Output Answer:** *An engine is part of a car.*

4.2 Data Preprocessing

Steps and Rationale:

- **Data Cleaning:**
  - *Remove Low-Confidence Edges:* Filter out edges with low confidence scores to reduce noise and improve the quality of retrieved triplets.
  - *Language Filtering:* Focus on English-language nodes and edges to simplify the initial implementation and evaluation.
  - *Normalization:* Normalize text to lowercase and remove special characters to ensure consistent matching and comparison.
- **Triplet Construction:**
  - *Extract Triplets:* Convert the cleaned data into triplets (subject, relation, object) for efficient retrieval.
  - *Indexing:* Create indices for fast lookup of triplets based on subjects and relations.
- **Embedding Generation:**
  - *Concept Embeddings:* Use a pre-trained language model (e.g., BERT) to generate embeddings for each concept (node) and relation.
  - *Storage:* Store these embeddings for quick similarity computation during question answering.

Rationale:

- **Quality Improvement:** Cleaning and filtering improve the reliability and relevance of the retrieved triplets.

- **Efficiency:** Preprocessing steps like triplet construction and embedding generation enable faster and more accurate retrieval during the QA process.

5 Baseline

1. **Entity-based Local Subgraph Retrieval**: This baseline utilizes entities in questions to retrieve local subgraphs from KGs. These subgraphs are then processed through a KG encoder, such as graph neural networks (GNNs), to model their local structures and integrate them into language models for question answering.

2. **Triplet Ranking with Language Models**: This method involves retrieving the most pertinent triplets from KGs and then reranking them. The reranked triplets are concatenated with the questions and input into language models.

3. **StandardGNN-based QA Model**: A conventional GNN-based model that encodes the entire KG and utilizes the encoded representations to respond to questions.

4. **Transformer-based QAModel**: A transformer-based model employing pretrained language models, like BERT or GPT, fine-tuned on the question answering task.

6 Results

The experimental results comparing the Knowledge Graph approach and the DREAM model on the Commonsense QA task are summarized below.

6.1 Knowledge Graph Approach

- **T5-BASED:** 0.3900

6.2 DREAM Model

- **T5-BASED:** 0.3604

- **T5-BASED with Elaboration:** 0.4046

6.3 Experimental Design

The experiments were conducted to evaluate the effectiveness of both approaches in enhancing the performance of the T5 model on the Commonsense QA task. The evaluation metric used was Accuracy@0.

Model	**Accuracy@0**
T5-BASED (Knowledge Graph)	0.3900
T5-BASED	0.3604
T5-BASED with Elaboration	0.4046

Table 1: Experimental Results

6.4 Results Table

6.5 Analysis

The results indicate that both approaches significantly enhance the performance of the T5 model. The Knowledge Graph approach achieves a base accuracy of 0.3900, and the DREAM model improves its performance with the addition of elaborations, achieving an accuracy of 0.4046. This highlights the effectiveness of context-rich explanations in improving model performance.

6.6 Discussion

The Knowledge Graph approach excels in efficiently retrieving relevant information and integrating it with dense vector representations, leading to a solid base accuracy. On the other hand, the DREAM model enhances the T5 model's performance through detailed contextual elaborations. Integrating both methodologies could potentially leverage the strengths of each approach, leading to even greater improvements in QA performance.

By combining the detailed processes and results, this document provides a comprehensive overview of the methodologies, experimental design, and outcomes of the Knowledge Graph and DREAM model approaches in enhancing the performance of LLMs on the Commonsense QA task.

7 Conclusion

This study compares two methodologies for enhancing the performance of Large Language Models (LLMs) on the Commonsense QA task: the Knowledge Graph approach and the DREAM-FLUTE method. The Knowledge Graph approach leverages Dense Passage Retrieval (DPR) and similarity comparisons with explanation embeddings to efficiently and accurately map questions to relevant knowledge from ConceptNet. The DREAM-FLUTE method integrates the DREAM model with T5-BASE and BERT models, utilizing detailed contextual elaborations to improve commonsense reasoning and figurative language understanding.

Experimental results demonstrate that both approaches significantly enhance the performance of the T5 model. The Knowledge Graph approach achieves a base accuracy of 0.3900, indicating its effectiveness in retrieving and integrating relevant information. The DREAM model, on the other hand, shows substantial improvement with the addition of elaborations, achieving an accuracy of 0.4046. This underscores the value of detailed contextual explanations in improving model performance.

The comparative analysis reveals that while the Knowledge Graph approach excels in efficient information retrieval and integration, the DREAM model enhances context understanding through elaborations. By leveraging the complementary strengths of both methodologies, there is potential for even greater improvements in QA performance. Future research could explore the integration of these approaches to develop more robust and accurate NLP models capable of advanced commonsense reasoning and figurative language understanding.

In conclusion, this study highlights the importance of combining structured knowledge representations with detailed contextual explanations to enhance the capabilities of LLMs. The findings provide valuable insights for advancing NLP research and developing more sophisticated models for a wide range of QA tasks.

8 Tools Used

In this project, the following tools and libraries were utilized:

- **BERT:** A pre-trained transformer model used for encoding questions and triplets.
- **T5-BASE:** A text-to-text transformer model used for generating and validating answers.
- **DPR (Dense Passage Retrieval):** Used for retrieving relevant passages based on dense vector similarities.
- **ConceptNet:** A knowledge graph used to provide structured knowledge for question answering.
- **Python:** The primary programming language used for developing and integrating various components.
- **PyTorch:** A deep learning framework used for implementing and fine-tuning models.
- **Hugging Face Transformers:** A library providing easy access to pre-trained transformer models like BERT and T5.
- **Scikit-learn:** Used for various machine learning utilities and evaluation metrics.

9 Code Repository

The code for this project can be found at the following GitHub repository:
https://github.com/hossein601/Knowledge-Extraction-from-Knowledge-Graphs-a

References

[Kar+20] Vladimir Karpukhin et al. “Dense Passage Retrieval for Open-Domain Question Answering”. In: *arXiv preprint arXiv:2004.04906* (2020).

[Mih+18] Todor Mihaylov et al. “Knowledgeable Reader: Enhancing Cloze-Style Reading Comprehension with External Commonsense Knowledge”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2018, pp. 821–831.

[SCH17] Rob Speer, Joshua Chin, and Catherine Havasi. “ConceptNet 5.5: An Open Multilingual Graph of General Knowledge”. In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017.

[Yih+15] Wen-tau Yih et al. “Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base”. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 2015, pp. 1321–1331.

[ZZW19] Duyu Zhang, Jianfeng Zhao, and Yi Wang. “Triple-Driven Neural Knowledge Base Question Answering”. In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI-19)*. 2019, pp. 4626–4632.