Efficient Common Sense in Large Language Models via Knowledge Graph Compression

Quentin Callahan qcallahan@ucsd.edu

Penelope King pking@ucsd.edu

Esther Cho ehcho@ucsd.edu

Yusu Wang yusuwang@ucsd.edu

Gal Mishne gmishne@ucsd.edu

Abstract

Commonsense knowledge is foundational for reasoning and decision making tasks, yet large language models (LLMs) can struggle with implicit knowledge. Knowledge graphs provide a rich source of structured commonsense relationships. However, the size and complexity of these graphs make traditional processing computationally expensive, particularly for capturing long-range dependencies. While graph neural networks (GNNs) are effective for graph-based tasks, they often struggle with scalability; transformers, capable of capturing long-range dependencies, scale poorly with graph size. A possible solution is provided by compressed knowledge graphs. Efficiently compressing knowledge graphs enables faster processing and training while also maintaining accuracy and performance of the final model. To address this, we propose an efficient knowledge graph compression method that selectively retains relevant commonsense relationships. By refining the structure of knowledge graphs, we aim to improve LLMs' ability to integrate commonsense knowledge for tasks such as semantic reasoning and abductive inference. Our approach uses a transformer-based architecture to enhance scalability and preserve essential long range relationships, maintaining key knowledge while ensuring diversity in generated outputs. Using datasets like ComVE and αNLG, we benchmark the effectiveness of our method while also improving computational efficiency. By bridging the gap between large-scale graph-based knowledge and LLMs, our work contributes to more efficient and context-aware commonsense reasoning in NLP applications.

Website: penelopeking.github.io/transformer-knowledge-graph-compression/
Code: github.com/PenelopeKing/Efficient-Common-Sense-in-LLMs-via-Knowledge-Graph-Compression

1 Introduction

Geometric deep learning is a subfield of deep learning that works with complex graph data structures. Geometric data often struggles with the complexity of how nodes are interconnected. Knowledge graphs are a type of graph data subset; they are structured representations of information where entities are nodes (people, places, or concepts) and their relationships are edges. Graph neural networks are a relevant technique to apply to this type of data for tasks such as node classification, community detection, and graph classification. From protein interactions, to social media networks, there are many applications for graph-based machine learning. However, many techniques still struggle with the complexity of large graph data, especially when it comes to knowing which long range interactions. This is where transformers come into play. Transformers are a type of deep learning architecture for sequential data, often used for natural language processing tasks. Their strengths lie in its ability for parallelization, having self-attention mechanisms, scalability, and versatility. Graphs–which are inherently non-sequential–may benefit from the unique advantages that transformers offer. In this project, we seek to explore transformers in a graph learning context, and see if it may have the potential to improve the successfulness of graph learning tasks. However, one issue that comes into play is how knowledge graphs in particular can be exceeding large, which becomes an issue especially since modern graph transformers tend to scale quadratically with the size of the graphs, making them infeasible on graphs beyond a certain scale.

Commonsense knowledge graphs (CSKGs) are a specialized type of knowledge graph designed to encode general world knowledge. They play a crucial role in various applications, including reasoning, decision-making, and natural language understanding. They can assist LLMs in generating commonsense explanations beyond what is explicitly mentioned in context. And compressing CSKGs can ensure that a LLM is fed concise knowledge without redundant or irrelevant information. Addressing this challenge involves developing methods that allow models to discern which concepts are essential, ensuring that the extracted knowledge is both meaningful and useful for downstream applications.

Work by Hwang et al. (2023) focuses on knowledge graph compression using a mixture of experts (MoE) model for generating commonsense explanations. The paper tackles the issue of the large size of knowledge graphs through a differentiable graph compression algorithm resulting in compressed subgraphs. With their methods, crucial concepts are able to be preserved for two commonsense generation tasks: commonsense explanation generation and abductive commonsense reasoning. Our work will largely base off of this study, focusing on transformers rather than an MoE based model for compression of CSKG subgraphs.

Our project aims to solve a similar problem using graph transformers as our method of compression. Due to a transformer's ability to capture long range dependencies and use of self attention, we predict that transformers would be a suitable method for compression on knowledge graphs. We similarly trial our methods on commonsense explanation generation and abductive commonsense reasoning, as well as comparing our methods to baselines and the MoE model created by Hwang et al. (2023).

Therefore, the question for our project will be: how effective are graph transformers for compressing knowledge graphs in maintaining the accuracy, diversity of output, and optimize speed of fitted models? Our hypothesis is that graph transformers will be able to capture long range interactions even on compressed graphs.

More work has been explored in relation to compression of knowledge graphs. Although there are few methods that involve transformers for knowledge graph distillation, there has been work done to compress high dimensional embeddings to low dimensional ones for knowledge graphs (Wang et al. 2021) (Yang et al. 2023). KGEs are a type of representation learning method that encodes knowledge graphs into a lower dimensional, continuous vector space. This change in structure allows for more efficient computation. Zhu et al. (2020) explores knowledge graph embeddings (KGE) for knowledge graph reasoning, specifically the high dimensionality of KGEs. This versatile method creates low-dimensional KGEs and considers the dual influence between teacher (high-dimensional) and student (low-dimensional) models. It incorporates a soft label evaluation mechanism to assign adaptive weights to different triples and employs a two-stage distillation process to enhance the student's assimilation of the teacher's knowledge.

1.1 Datasets

The data we used consisted of two CSKGs: ComVE and αNLG. In ComVE (Wang et al. 2020) the goal is to generate explanations on why a nonsensical sentence does not make sense. Each sample comes with a 3 reference output sentence, which are human written explanations. The dataset has a training size of 10k, and a test and validation size of 1000. On the other hand, for α-NLG (Bhagavatula et al. 2020) the task is to generate a plausible explanation for what might have happened in between a past and future observation, which is also known as adjustive reasoning. Each sample in the dataset includes up to 5 reference outputs. This dataset has 50k training points, over 1,500 validation points, and over 3,500 test data points.

2 Methods

2.1 Problem Formulation

Following Hwang et al. (2023) we aim to improve the quality and diversity of transformer language models on generative commonsense reasoning tasks such as commonsense explanation generation and abductive commonsense reasoning, by leveraging a Transformer-based model in place of the relational graph convolutional network (r-GCN). Given an input sequence x, our goal is to model a conditional distribution $p(y|x)$ that assigns high probabilities to multiple target outputs $y_1, ..., y_K$, ensuring both diversity and coherence.

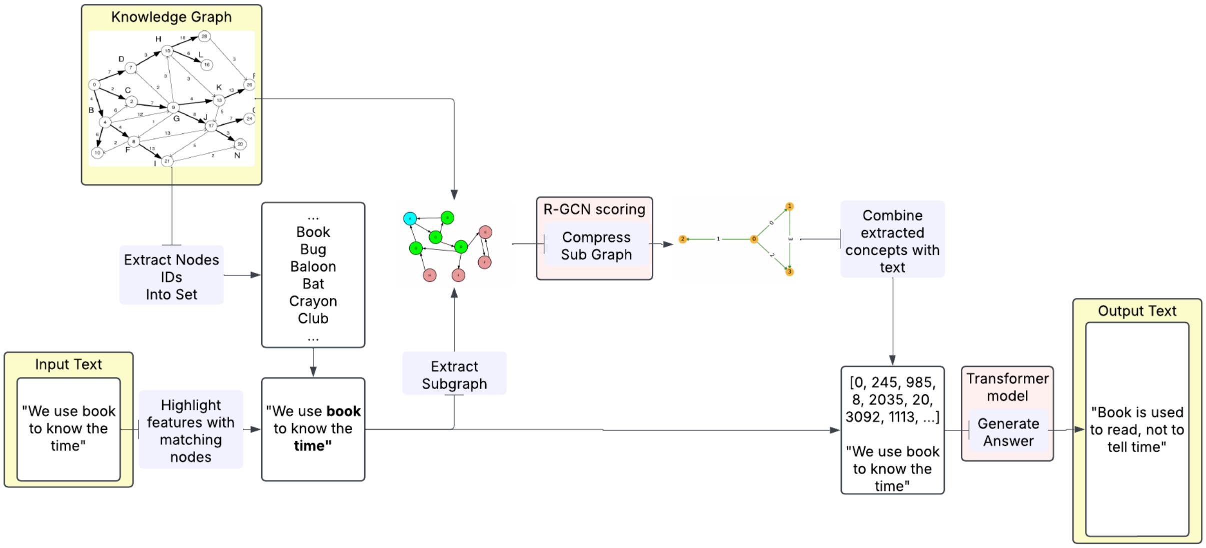
Knowledge Graph
Extract Nodes IDs Into Set
...
Book
Bug
Baloon
Bat
Crayon
Club
...
R-GCN scoring
Compress Sub Graph
Combine extracted concepts with text
Input Text
"We use book to know the time"
Highlight features with matching nodes
"We use **book** to know the **time"**
Extract Subgraph
[0, 245, 985, 8, 2035, 20, 3092, 1113, ...]
"We use book to know the time"
Transformer model
Generate Answer
Output Text
"Book is used to read, not to tell time"

Figure 1: Architecture of Hwang et al. (2023)

2.2 Model Overview

Figure 1 shows a high-level overview of the architecture proposed by Hwang et al. (2023), where an input text is used to extract a relevant subgraph, then an R-GCN model is used to compress the subgraph. The compressed subgraph is transformed into a list of concept ids, which are fed into a transformer model along with the initial input text to generate a final output based on a given task.

Our model consists of two primary deviations from this architecture:

1. **Transformer-based Knowledge Graph Encoding:** Instead of using R-GCN to encode knowledge graph (KG) information, we use a graph transformer model to make better use of global graph information to determine which nodes will be important for a given task.
2. **Text injection** instead of using concept node ids as input to the transformer language model, we convert the relationships to text to avoid discarding useful edge information.

2.3 KG Subgraph Extraction

To process the commonsense knowledge graphs (CSKG) for our experiments, we used the subgraph extraction by Hwang et al. (2023). This phase is important because it keeps the contextual information required for these reasoning tasks while lowering the computational cost.

By directly matching words to node labels in the CSKG, we are able to extract important ideas from an input phrase. For example, the sentence “A person cannot walk across water because water is not solid" we can extract the main concepts like " person," "walk," "water," and "solid" which would serve as the seed nodes for the subgraph extraction. C_q = person, walk, water, solid. Once these key concepts are identified, we can expand outward to retrieve their neighboring nodes within a fixed number of hops. We define a radius h which determines how far we expand from the concepts. If h = 2, we include all nodes up to 2 hops away from the concepts in C_q. The paper Hwang et al. (2023) uses h = 2 for their model. For our experiments, we use: h = 1, h = 2, and h = 3 to compare performance on larger and smaller subgraphs. We hypothesize transformer based compression will yield a more significant improvement at higher h values.

2.4 Multi-relational Graph Encoding

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$ denote the extracted subgraph from the commonsense knowledge graph, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of edges, and $\mathcal{R}$ is the set of relation types. The model from Hwang et al. (2023) employs a relational graph convolutional network (R-GCN) to encode this graph. Specifically, let $h_v^{(l)} \in \mathbb{R}^d$ be the embedding of node v at layer l, and let $h_r^{(l)}$ be the embedding of relation r at layer l. We update each node v by first aggregating its neighbors' representations:

$$o_v^{(l)} = \frac{1}{\left|\mathcal{N}(v)\right|} \sum_{(u,v,r)\in\mathcal{E}} W_N^{(l)}\, \phi\left(h_u^{(l)}, h_r^{(l)}\right), \tag{1}$$

where $\mathcal{N}(v)$ denotes the set of neighbors of v, and $W_N^{(l)}$ is a learnable parameter matrix. The compositional function $\phi(\cdot)$ fuses the node embedding $h_u^{(l)}$ with the relation embedding $h_r^{(l)}$; we define

$$\phi\left(h_u^{(l)}, h_r^{(l)}\right) = h_u^{(l)} - h_r^{(l)},$$

We then obtain the next-layer embedding of v via

$$h_v^{(l+1)} = \mathrm{ReLU}\left(o_v^{(l)} + W_S^{(l)} h_v^{(l)}\right). \tag{2}$$

Here $W_S^{(l)}$ is a learnable parameter matrix. Meanwhile, we also update the relation embeddings themselves by

$$h_r^{(l+1)} = W_R^{(l)} h_r^{(l)}, \tag{3}$$

where $W_R^{(l)}$ is another learnable parameter matrix for relation r.

After the final R-GCN layer, each node's representation $h_i^{(L)}$ is passed through a multi-layer perceptron (MLP) to obtain a scalar score:

$$s_i = \mathrm{MLP}(h_i^{(L)}). \tag{4}$$

The top N nodes with the highest scores s_i are then selected as inputs to the language model transformer.

While this method enables each node's embedding to capture local structural information, it does not directly incorporate global context. To address this, we replace the R-GCN with a graph transformer architecture that alternates between local R-GCN layers and global attention layers.

For the global attention layer, we first compute the attention weights between any two nodes i and j as

$$\alpha_{ij} = \frac{\exp\left(\left(W_Q h_i\right)^\top \left(W_K h_j\right) / \sqrt{d}\right)}{\sum_{k\in\mathcal{V}} \exp\left(\left(W_Q h_i\right)^\top \left(W_K h_k\right) / \sqrt{d}\right)}, \tag{5}$$

where W_Q, W_K, and W_V are learnable projection matrices for the query, key, and value, respectively, and d is the dimensionality of the node embeddings. The updated representation from the attention mechanism is then given by

$$h_i^{\text{att}} = \sum_{j\in\mathcal{V}} \alpha_{ij} W_V h_j. \tag{6}$$

This attention output is combined with the original node features using a residual connection and layer normalization, to ensure information gained from local features is preserved:

$$h_i^{(l+1)} = \text{LayerNorm}\left(h_i^{(l)} + h_i^{\text{att}}\right). \tag{7}$$

This combination of local (R-GCN) and global (attention or global node) mechanisms enables each node's embedding to reflect both its immediate relational context and the global structure of the subgraph, which we hypothesize leads to more effective node scoring and improved downstream performance.

In addition to the graph transformer, we explore alternative methods for incorporating global information into the R-GCN. One approach adds random long-range connections to the subgraph, where the new edges are assigned their own distinct edge type so that the model can distinguish them from the original relations. Another approach augments the graph with a globally connected hidden node.

2.5 Loss

Similar to the work done by Hwang et al. (2023) and Yu et al. (2022), we are training BART-based (Lewis et al. 2020) using a seq2seq architecture for the commonsense explanation generation task. The loss metrics we use when training are generation loss, KG concept loss, and optimal transport loss.

Generation Loss. Generation loss measures how well the generated commonsense explanation aligns with the ground truth explanation. We use a cross-entropy loss over the sequence of tokens:

$$\mathcal{L}_{\text{gen}} = -\sum_{t=1}^{T} \log P(y_t \mid y_{<t}, x) \tag{8}$$

where x is the input (e.g., a knowledge graph representation or a natural language prompt), y_t is the target token at time step t, and $P(y_t \mid y_{<t}, x)$ is the probability assigned by the

model to the correct token given the previous sequence. The loss is summed over the sequence length T.

KG Concept Loss. To encourage the model to attend to relevant knowledge graph (KG) concepts, we introduce a KG concept loss. This ensures that the model correctly selects concepts from the knowledge graph that contribute to the explanation. Let $\hat{C}$ be the predicted set of selected KG concepts and C be the ground truth concept set. We define the KG concept loss using a binary cross-entropy (BCE) objective:

$$\mathscr{L}_c = -\left(\sum_{c \in V_q \cap C_a} y_c \log P(c) + \sum_{c \in V_q - C_a} (1 - y_c) \log(1 - P(c)) \right) \tag{9}$$

where c_i is a binary indicator for whether the i-th concept should be included, and $\hat{c}_i$ is the predicted probability of including that concept.

Optimal Transport Loss. To make the optimal transport distance differentiable, we solve it using Sinkhorn's algorithm (Cuturi 2013). As defined by Hwang et al. (2023), the optimal transport loss is approximated by:

$$\mathscr{L}_t = W_\gamma^k(G, G_c) = \langle P^k, M \rangle - \gamma E(P^k) \tag{10}$$

where $W_\gamma^k(G, G_c)$ is the entropic-regularized Wasserstein distance (Sinkhorn distance), measuring the difference between the graphs G and G_c. The term P^k represents the transport plan after k iterations of the Sinkhorn algorithm, while M is the cost matrix defining distances between nodes in G and G_c. The transport cost is given by $\langle P^k, M \rangle = \sum_{i,j} P_{ij}^k M_{ij}$, and $E(P^k) = \sum_{i,j} P_{ij}^k \log P_{ij}^k$ is the entropy of the transport plan. The parameter γ controls the level of entropy regularization, ensuring a smooth and numerically stable optimization. The first term minimizes transport cost, while the second term encourages a well-structured transport plan.

Overall Loss. The final loss function combines all three components with weighting coefficients λ_{KG} and λ_{OT}:

$$\mathscr{L} = \mathscr{L}_{\text{gen}} + \lambda_{\text{KG}} \mathscr{L}_{\text{KG}} + \lambda_{\text{OT}} \mathscr{L}_{\text{OT}} \tag{11}$$

where λ_{KG} and λ_{OT} are hyperparameters that balance the contributions of KG concept loss and optimal transport loss, respectively.

2.6 Metrics

We evaluate the performance of the models based on 3 ideas: pairwise diversity, corpus diversity, and quality. These evaluations are based on past work done by Hwang et al. (2023) and similar works.

Pairwise Diversity. Self-BLEU (Zhu et al. 2018) evaluates how each sentence is similar to other generated sentences based on n-gram overlap. We are using 2 types of self-BLEU metrics: self-BLEU 3–which looks at 3-gram overlap–and self-BLEU 4–which looks at 4-gram overlap. A lower self-BLEU scores indicates that there is greater variety between sentences in the set generated for each input sample

Corpus Diversity. Entropy-k (Zhang et al. 2018), evaluates evenness of empirical n-gram distribution within generated sentences. And distinct-k (Li et al. 2016) is calculated by counting the number of unique k-grams in generated sentences and dividing it by the total number of generated tokens. This metric helps prevent preference towards longer sentences.

Quality. Quality metrics assess the highest accuracy by comparing the best generated sentences to the target sentences. This is evaluated using BLEU (Papineni et al. 2002) and ROUGE (Lin 2004), which both are used for n-gram overlap between generated sentences and human-written reference outputs.

3 Results

3.1 Baseline Performance

3.2 Transformer Performance

4 Discussion

5 Conclusion

6 Contributions

Quentin Callahan: Data preprocessing and subgraph extraction pipeline, compression techniques code, and writing methods in report draft.

Esther Cho: Writing comparison code from Hwang paper, research into CSKG data– specifically edge relations for decoding, and organizing and formatting code for code checkpoint.

Penny King: Writing report draft, creating website, score metric code, and in charge of meetings, timeline, reports and slides.

References

Bhagavatula, Chandra, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen tau Yih, and Yejin Choi. 2020. “Abductive Commonsense Reasoning." In *International Conference on Learning Representations*. [Link]

Cuturi, Marco. 2013. "Sinkhorn Distances: Lightspeed Computation of Optimal Transportation Distances." [Link]

Hwang, EunJeong, Veronika Thost, Vered Shwartz, and Tengfei Ma. 2023. "Knowledge Graph Compression Enhances Diverse Commonsense Generation." In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Singapore Association for Computational Linguistics. [Link]

Lewis, Mike, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online Association for Computational Linguistics. [Link]

Li, Jiwei, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. "A Diversity-Promoting Objective Function for Neural Conversation Models." In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California Association for Computational Linguistics. [Link]

Lin, Chin-Yew. 2004. "ROUGE: A Package for Automatic Evaluation of Summaries." In *Text Summarization Branches Out*. Barcelona, Spain Association for Computational Linguistics. [Link]

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. "BLEU: a method for automatic evaluation of machine translation." In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. USA Association for Computational Linguistics. [Link]

Wang, Cunxiang, Shuailong Liang, Yili Jin, Yilong Wang, Xiaodan Zhu, and Yue Zhang. 2020. "SemEval-2020 Task 4: Commonsense Validation and Explanation." In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online) International Committee for Computational Linguistics. [Link]

Wang, Kai, Yu Liu, Qian Ma, and Quan Z. Sheng. 2021. "MulDE: Multi-teacher Knowledge Distillation for Low-dimensional Knowledge Graph Embeddings." *Proceedings of the Web Conference 2021*: 1716–1726. [Link]

Yang, Zhendong, Ailing Zeng, Zhe Li, Tianke Zhang, Chun Yuan, and Yu Li. 2023. "From Knowledge Distillation to Self-Knowledge Distillation: A Unified Approach with Normalized Loss and Customized Soft Labels." 07. [Link]

Yu, Wenhao, Chenguang Zhu, Lianhui Qin, Zhihan Zhang, Tong Zhao, and Meng Jiang. 2022. "Diversifying Content Generation for Commonsense Reasoning with Mixture of Knowledge Graph Experts." In *Findings of the Association for Computational Linguistics: ACL 2022*. Dublin, Ireland Association for Computational Linguistics. [Link]

Zhang, Yizhe, Michel Galley, Jianfeng Gao, Zhe Gan, Xiujun Li, Chris Brockett, and Bill Dolan. 2018. "Generating Informative and Diverse Conversational Responses via Adversarial Information Maximization." [Link]

Zhu, Yaoming, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. "Texygen: A Benchmarking Platform for Text Generation Models." [Link]

Zhu, Yushan, Wen Zhang, Mingyang Chen, Hui Chen, Xu Cheng, Wei Zhang, and Hua-

jun Chen. 2020. “DualDE: Dually Distilling Knowledge Graph Embedding for Faster and Cheaper Reasoning.” [Link]

Appendices

Project Proposal:
https://drive.google.com/file/d/14NA03_Kc2cdHmtRLu5m83iipOkVG-AVv/view?usp=sharing