6-18-2022

Can Language Models Capture Graph Semantics? From Graphs to Language Model and Vice-Versa

Tarun Garg

Kaushik Roy

Amit Sheth

CAN LANGUAGE MODELS CAPTURE GRAPH SEMANTICS? FROM GRAPHS TO LANGUAGE MODEL AND VICE-VERSA

Tarun Garg
BITS Pilani
f20160450h@alumni.bits-pilani.ac.in

Kaushik Roy
AI Institute, University of South Carolina
kaushikr@email.sc.edu

Amit Sheth
AI Institute, University of South Carolina
amit@sc.edu

ABSTRACT

Knowledge Graphs are a great resource to capture semantic knowledge in terms of entities and relationships between the entities. However, current deep learning models takes as input distributed representations or vectors. Thus, the graph is compressed in a vectorized representation. We conduct a study to examine if the deep learning model can compress a graph and then output the same graph with most of the semantics intact. Our experiments show that Transformer models are not able to express the full semantics of the input knowledge graph. We find that this is due to the disparity between the directed, relationship and type based information contained in a Knowledge Graph and the fully connected token-token undirected graphical interpretation of the Transformer Attention matrix.

Keywords Knowledge Graph, Graph Neural Networks, Transformers

1 Introduction

Natural Language is a widely researched domain focusing on interpreting and processing human language for better human-machine interaction. Taking insights from Natural Language is challenging as most data is unstructured and cannot be fed directly into the machine. Models like BERT **Devlin et al. [2018]**, GPT **Brown et al. [2020]** are increasingly being used to solve major Natural Language problems. Question Answering, Machine translation, and Information retrieval are examples that use these underlying algorithms. In Natural Language research, various forms of embedding like Glove **Pennington et al. [2014]**, Word2Vec **Mikolov et al. [2013]**, TF-IDF **Ramos et al. [2003]** are used to vectorize the textual data. The vector embeddings are then passed over as an input to the Machine learning Models.

Knowledge Graphs are an emerging form of Knowledge capture that represents structured semantic knowledge widely available on the web. Knowledge Graphs is a form of a Graphical Database incorporating domain knowledge resulting in scalable and efficient usability. KGs have been used in the past with Deep Networks to develop algorithms such as Graph Attention Networks **Veličković et al. [2017]**, OpenKE **Han et al. [2018]**, Graph Convolution Transformer **Choi et al. [2019]** which are guided by the structured knowledge to make better predictions.

Use of Knowledge Graphs in language models had been done in the past for various applications like QA-GNN **Yasunaga et al. [2021]**, Entity2rec **Palumbo et al. [2017]**. They have successfully managed to use KG to improve the performance of Language models. Despite the improved performance, as such methods compress the KG into vectorized representation, these representations haven't yet been demonstrably shown to capture the full semantics information in KG.**Swamy et al. [2021]**, **Jain et al. [2021]**.

Considering many applications of Knowledge Graph in Natural Language, we chose to conduct a study to figure out whether Language Models have the semantic capability to store Knowledge Graphs? Here we have performed some

experiments to understand it and provided insights and detailed discussion on the possible shortcomings. We have tried to provide adequate pointers to allow further research on this.

2 Background

2.1 Knowledge Graphs

As mentioned above, Knowledge graphs are Graphical Database being widely used in Natural Language to improve the existing models through techniques such as Knowledge infusion **Sheth et al. [2021]**, **Kursuncu et al. [2019]**, **Su et al. [2021]**. The use of KG is attractive because of the wide variety of different kinds of knowledge that they capture online. Some of the widely used KGs available on the web are as follows:

- Wikidata **Vrandečić and Krötzsch [2014]** is one of the largest, freely available Knowledge Graph, which organizes data in a structured graphical format for better information extraction. It is a multilingual data prepared majorly by Wikipedia and other Wikimedia sister projects. Wikidata editors maintain the data, and the items are encoded with a unique key starting with Q and followed by a number.
- ConceptNet is another freely-available semantic network that stores data relating to the words people use and their semantic meaning. The data is prepared from various crowd-sourced projects. This network can be used to help computers understand words from the natural language humans commonly use by creating word embeddings. **Speer et al. [2017]**.
- DBpedia is a crowd-sourced initiative to extract structured content from the information present in various Wikipedia resources. DBpedia is multilingual and represents actual community agreement. It regularly evolves as the content on Wikipedia is updated **Auer et al. [2007]**.

2.2 Language Models

Language models use deep learning methods to determine the probability of a given sequence of words occurring in a sentence **Lavrenko and Croft [2017]**. Language models started with sequence models using RNNs **Cho et al. [2014]**, LSTMs **Hochreiter and Schmidhuber [1997]**, and GRUs **Cho et al. [2014]**. Attention models were discovered that could help the model *focus* on particular parts of the input. Transformer models utilize self attention to learn word and token level correlations while performing predictions **Vaswani et al. [2017]**. Recent research has used transformer technology to combine graphs and language models to predict the output.

2.3 Graph Neural Networks - GNN

Graph Neural Networks **Scarselli et al. [2008]** is a differentiable message passing method that aggregates information from neighbouring nodes into node representations using convolution operations. However, they do not consider relationship information. R-GCN(Relational Graph Convolution Networks) **Schlichtkrull et al. [2018]** attempt to incorporate relationship information on the links enforcing sparsity and regularization constraints while constructing node representations. However, real-world knowledge graphs have few relationships and thus require modulation of a graph node network that does not contain relations with a graph that contains few relations between the same nodes. Thus, we employ the Graph Convolutional Transformer **Choi et al. [2019]** to achieve this modulation.

2.4 Graph Convolution Transformer - GCT

Graph Convolution Transformer **Choi et al. [2019]** uses the prior probabilities calculated from the domain knowledge graph to train the model. The model architecture consists of 3 transformer stacks with 2 feed forward layers. eICU Collaborative Research Database **Pollard et al. [2018]** consisting of anonymized medical details of patients is used to train the model.

GCT relies on the intuition that the Attention matrix in a Transformer can be thought of as a fully connected weighted graph among tokens. Thus, allowing for integration between the two graphical structures (Knowledge Graph and the Attention matrix graph). However, the nature of the graphs are very different. One is a directed graph with relationships and schema (type) information and the other is a fully connected weighted undirected graph. As we will see in our experiments, this disparity will be an impediment in allowing adequate semantic knowledge capture from the Knowledge Graph by the Transformer architecture.

(a) Example path followed to generate triple

Step	Action	Possible intermediates	Match
0	START	(Sam,	0
1	YIELD	(Sam,is,	0.4
2	YIELD	(Sam,is,footballer	0.9
3	STOP	(Sam,is,footballer)	0.9

(b) Procedure to generate a triple for the given context

		Query			
		Sam	is	a	footballer
Key	Sam	x	x	x	x
	is	0.4	x	x	x
	a	0.1	0.2	x	x
	footballer	0.2	0.5	0.1	x

(c) Resulting attention matrix from the sample example

Figure 1: An illustration showing the procedure to generate the triple

GCT working relies on 2 major components: i) KL-divergence ii) negative log-likelihood. KL Divergence helps preserve the domain knowledge, and log-likelihood helps find new relations for making predictions. Here we have used GCT and tried some modifications to quantify the domain knowledge captured in the Language Model.

2.5 Language models are open knowledge graphs

Previous models have tried to illustrate that language models do capture information present in KG. For example one such methodology to generate Knowledge Graph from Language Models was proposed in 2020 **Wang et al. [2020]**, which incorporates the concept of Conditional Probability. Here the researchers have proposed that the attention matrix can be used as the conditional probability, which can further be used to construct a knowledge graph. Attention weights denote the correlation of one word with another which can be used to label the corresponding entities and relations, leading to the generation of a new Knowledge Graph.

In the sample matrix (Figure 1) [1], the author has tagged an entity and predicted the next terms of the triple based on conditional probability extracted from the attention matrix. In the example shown, the algorithms start with the word 'Sam'. Corresponding to 'Sam', the word with maximum attention is 'is' and similarly 'footballer' will be the next word corresponding to 'is'.

However, there are various restrictions on how this method this method generates the KG. Notably all KG triples are not forward directional and schema level information that is essential to disambiguate entities in different context is not at captured. This schema-level information that contains various contextual relationships represents the heart of the semantics in the KG required for reasoning.

3 Experiments

The question remains can we generate the full semantics of a Knowledge Graph from Language Models? We test the hypothesis that does the Graph Convolution Transformer **Choi et al. [2019]** has enough expressivity to encode the causal context in the knowledge graph. If the predicted structure matches the knowledge graph that is encoded in the

[1] Figures inspired by paper Wang et al. [2020]

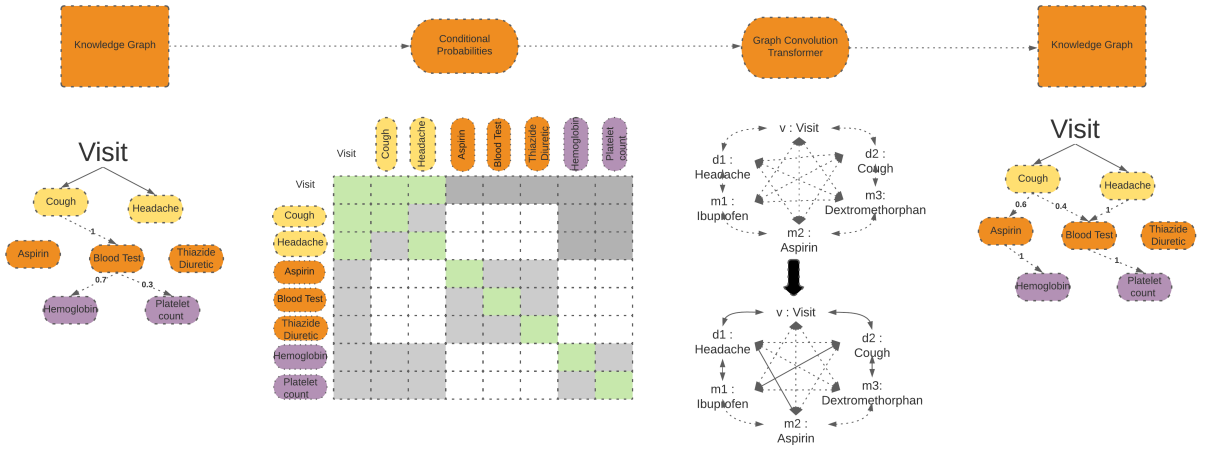

Figure 2: (a) The first transition transition talks about calculating conditional probabilities based on the occurrence of entity relation pairs. This results in the prior attention matrix.(b) Second transition shows the relational mapping of the context through Graph Convolution Transformer. The darker edged signifies strong relationship between the linked entities. (c) Attention matrix is then used to reproduce the Knowledge Graph based on the probabilities

Graph Convolution Transformer (GCT), the hypothesis is confirmed. If a model such as GCT can capture a specific context such as causal, we may hope that other context representing the full semantics of a KG can be captured in future Language Models.

GCT working involves reading data across various files including diagnosis, treatment details and are grouped based on patient ID. Conditional probabilities are then calculated for Diagnosis-procedure and procedure-diagnosis pairs. The grouped data is then stored as TfRecord, which is then processed by the training module.

Conditional probabilities are generated based on the occurrences of the terms together for a particular patient. These conditional probabilities are marked as prior probabilities used as the non-trainable attention weights of the first transformer stack. Further layers are trained using KL divergence and sigmoid cross-entropy as the loss function to preserve the domain knowledge by penalizing the difference between the attention weights of consecutive layers. We will be using the attention weights generated in the final transformer stack to develop a new graph.

We encode the graph structure into the graph neural network through a convolution operation to modulate the graph node representations (GCT). Next, we tried to recover the graph structure by predicting the graph structure from the language model (Language Models are open knowledge graphs) (see **Figure** 2).

To understand the importance of GCT architecture, various changes were made in architecture of GCT to measure the effect of each layer. Firstly, condition probabilities of each procedure-disease and disease-procedure relation are calculated and stored separately for each patient. While passing the data to Graph Convolution Transformer. The condition probabilities are pushed as the first attention matrix. KL Divergence is used further along with cross entropy loss to train the model. Note that the key difference from the original GCT paper is that we impose the KL divergence minimization at every transformer block.

KL Divergence Formula

$$Define\hat{A}^{(j)} := softmax(\frac{Q^{(j)}K^{(j)T}}{\sqrt{d}} + M) \tag{1}$$

$$Self - Attention :$$

$$C^{(j)} = MLP^{(j)}(PC^{(j-1)}W_v^{(j)})when j = 1, \tag{2}$$

$$C^{(j)} = MLP^{(j)}(\hat{A}^{(j)}C^{(j-1)}W_V^{(j)})when j > 1$$

$$Regularization:$$

$$L_{reg}^{(j)} = D_{KL}(P||\hat{A}^{(j)})when j = 1, \tag{3}$$

$$L_{reg}^{(j)} = D_{KL}(\hat{A}^{(j-1)}||\hat{A}^{(j)})when j > 1$$

$$OrginalLossFunction:$$

$$L = L_{(pred)} + \lambda \sum_{j} L_{reg}^{(j)} \tag{4}$$

$$ModifiedLossFunction:$$

$$L = \lambda \sum_{j} L_{reg}^{(j)} \tag{5}$$

To experiment the impact of KL-divergence **Joyce [2011]** on Language Models, we made a change in the loss function of GCT. We removed the cross entropy loss from loss function and allowed the model to train on eICU data **Pollard et al. [2018]** with only KL-divergence. Our hypothesis in this experiment was that with only KL-divergence, the loss between layers should significantly decrease as it will try to preserve maximum possible information. At the same time AUC-PR and AUC-ROC should decrease significantly because it is no more using feedback from the labels. After training with new loss function, we tried reproducing the original knowledge graph.

4 Results

Modified Loss function				**Original Loss function**			
Steps	AUC-PR	AUC-ROC	loss	Steps	AUC-PR	AUC-ROC	loss
100	0.091	0.354	0.113	100	0.136	0.569	0.817
200	0.082	0.28	0.112	200	0.122	0.517	1.09
300	0.081	0.277	0.112	300	0.148	0.599	1.013
400	0.079	0.254	0.113	400	0.127	0.519	1.129
500	0.078	0.236	0.112	500	0.118	0.497	1.297
600	0.081	0.273	0.112	600	0.122	0.51	1.29
700	0.079	0.249	0.112	700	0.139	0.576	1.345
800	0.079	0.255	0.112	800	0.109	0.464	1.128
900	0.081	0.273	0.112	900	0.097	0.408	1.302
1000	0.08	0.264	0.112	1000	0.102	0.439	1.525
1100	0.078	0.246	0.112	1100	0.105	0.449	1.311
1200	0.079	0.251	0.112	1200	0.109	0.466	1.314
1300	0.077	0.239	0.112	1300	0.097	0.408	1.449
1400	0.081	0.278	0.112	1400	0.099	0.418	1.505
1500	0.08	0.268	0.112	1500	0.105	0.449	1.557
1600	0.083	0.299	0.112	1600	0.105	0.449	1.568
1700	0.083	0.3	0.112	1700	0.153	0.608	0.941
1800	0.084	0.313	0.112	1800	0.125	0.507	1.14
1900	0.086	0.322	0.112	1900	0.14	0.534	1.407
2000	0.086	0.328	0.112	2000	0.145	0.548	1.332

Table 1: Precision, Recall and loss values with edited loss-function (without log-likelihood, only KL-divergence) and original loss function (KL-divergence with log-likelihood). Original loss functions keeps track of the labels whereas with edited loss function, we have tried to capture the information retention of Knowledge Graphs in Language Models.

Experiment results (see **Table** 1) showed our hypothesis to be true. With the modified loss-function, the AUC-ROC and AUC-PR values dropped significantly. It was expected since we were not penalising wrong predictions anymore.

Loss between attention layers also decreased as expected. The new loss was less than 10% of the loss with original loss function. Still, the graph reproduced was not able to capture the underlying semantics of the input graph.

For generating Knowledge graphs we need an attention matrix. Once the attention matrix is generated, they are used with the existing pair of entities provided to find the path between those entities. The conditional probability of occurrence of each entity after a entity is used to determine the next entity. Once we have the path between two extreme entities, we combine them to define a sequence.

This entire process involves taking Knowledge Graph as an input feature $->$ Generating attention matrix using conditional probabilities $->$ Regeneration of Knowledge from trained attention matrix.

5 Conclusion

It is evidenced that due to the disparity in Knowledge Graphs (directed with relationships) and the fully connected graph interpretation of transformer architecture, graphical structure is imposed through KL divergence at all the layers, cannot capture the graphical structure adequately. Thus,in cases when we are trying to predict relations from an attention matrix in the general case, it is quite possible that there are multiple relations. Thus taking only max attention value eliminates the other relations. An alternative could be to use a threshold value instead of max value, but choosing that threshold value is itself a challenging task. Also in case of incomplete data where there's no KG triplet, it will tend to predict unwanted relations as the attention weights will still be significant. Thus the idea of graph structures from Language Models needs further research to overcome the limitations prevailing in the current models. Language models although excellent at GLUE tasks show significant limitations in semantic knowledge capture ultimately required for more intelligent systems, for example, to enable common sense reasoning.

6 Future Work

Knowledge-infused learning is an emerging field. A lot of research is going on to improve the existing models by infusing a knowledge base to provide better results.

Future research would be to restructure neurosymbolic approaches or graph neural network and similar approaches to construct a model which would be flexible with incorporating the richness of the graph structures. Constructing graphs from Language Models still requires a lot more research to work well with incomplete/extremely longer sentences and any other form of data which can't be transformed to a sentence.

To summarize, here we have discussed the conclusions derived while formulating the idea. More research needs to be conducted in future to convert the idea into a working model. The direction would be to experiment with other models to further understand the boundaries of large transformer models and how they can be stretched.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer, 2003.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Xu Han, Shulin Cao, Xin Lv, Yankai Lin, Zhiyuan Liu, Maosong Sun, and Juanzi Li. Openke: An open toolkit for knowledge embedding. In *Proceedings of the 2018 conference on empirical methods in natural language processing: system demonstrations*, pages 139–144, 2018.

Edward Choi, Zhen Xu, Yujia Li, Michael W Dusenberry, Gerardo Flores, Yuan Xue, and Andrew M Dai. Graph convolutional transformer: Learning the graphical structure of electronic health records. *arXiv preprint arXiv:1906.04716*, 2019.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. *arXiv preprint arXiv:2104.06378*, 2021.

Enrico Palumbo, Giuseppe Rizzo, and Raphaël Troncy. Entity2rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation. In *Proceedings of the eleventh ACM conference on recommender systems*, pages 32–36, 2017.

Vinitra Swamy, Angelika Romanou, and Martin Jaggi. Interpreting language models through knowledge graph extraction. *arXiv preprint arXiv:2111.08546*, 2021.

Nitisha Jain, Jan-Christoph Kalo, Wolf-Tilo Balke, and Ralf Krestel. Do embeddings actually capture knowledge graph semantics? In *European Semantic Web Conference*, pages 143–159. Springer, 2021.

Amit Sheth, Manas Gaur, Kaushik Roy, and Keyur Faldu. Knowledge-intensive language understanding for explainable ai. *IEEE Internet Computing*, 25(5):19–24, 2021.

Ugur Kursuncu, Manas Gaur, and Amit Sheth. Knowledge infused learning (k-il): Towards deep incorporation of knowledge in deep learning. *arXiv preprint arXiv:1912.00512*, 2019.

Yusheng Su, Xu Han, Zhengyan Zhang, Yankai Lin, Peng Li, Zhiyuan Liu, Jie Zhou, and Maosong Sun. Cokebert: Contextual knowledge selection and embedding towards enhanced pre-trained language models. *AI Open*, 2:127–134, 2021.

Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

Robyn 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.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer, 2007.

Victor Lavrenko and W Bruce Croft. Relevance-based language models. In *ACM SIGIR Forum*, volume 51, pages 260–267. ACM New York, NY, USA, 2017.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer, 2018.

Tom J Pollard, Alistair EW Johnson, Jesse D Raffa, Leo A Celi, Roger G Mark, and Omar Badawi. The eicu collaborative research database, a freely available multi-center database for critical care research. *Scientific data*, 5 (1):1–13, 2018.

Chenguang Wang, Xiao Liu, and Dawn Song. Language models are open knowledge graphs. *arXiv preprint arXiv:2010.11967*, 2020.

James M Joyce. Kullback-leibler divergence. *International encyclopedia of statistical science*, 720:722, 2011.