IAKG: Importance-augmented Knowledge Graphs

Stanford CS224N Custom Project

Jungmin "Josh" Cho
Department of Computer Science
Stanford University
joshcho@stanford.edu

Abstract

In recent years, Pre-trained Language Models (PLMs) have seen a meteoric rise, largely because through their training process, PLMs form a rich, implicit representation of domain knowledge. Yet its implicit nature makes knowledge inscrutable, which calls the need for explicit representations like Knowledge Graphs (KGs). KGs, while powerful in representing complex relationships between entities, do not capture priority between triples. In this paper, we propose a method for augmenting KGs with importance weights using Wikipedia as proxy for importance. As Wikipedia is limited as a dataset, we use PLMs to generalize the notion of importance from the Wikipedia dataset. We see importance-augmented knowledge graph as an important step in constructing richer explicit representations of knowledge.

1 Background

Knowledge graphs (KGs) are a popular and effective way to represent structured knowledge. They are composed of entities (nodes) and relationships (edges) between those entities. KGs have been used in various tasks such as question answering, recommendation systems, and information retrieval.

1.1 Limitations of Knowledge Graphs

Formally a knowledge graph $\mathcal{KG}$ can be defined as:

$$\mathcal{KG} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$$

where $\mathcal{E}$ is the set of entities (e.g. Einstein, Germany, Theory of Relativity) and $\mathcal{R}$ is the set of relations (e.g. born in, discovered). $\mathcal{T}$ is the set of triples in the form of (h, r, t) with h as head, t as tail, and r as relation, where $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$. Thus for the fact "Einstein is born in Germany", $h =$ Einstein, $r =$ born, and $t =$ Germany.

If a knowledge graph is to be a proxy for how knowledge rests in the mind, then a knowledge graph — specifically a knowledge graph triple — is insufficient at representing factoids in the mind. Consider the example above with Einstein. The two facts "Einstein is born in Germany" and "Einstein discovered theory of relativity" are represented as triples in the knowledge graph. Yet consider that to a human mind, Einstein's discovery of theory of relativity is more "important" than his birthplace. Knowledge graph representation fails to capture the priority between these two factoids with which most humans would agree.

An immediate counterpoint is the subjectivity of "importance". To some minds (perhaps a student studying the nationality of prominent scientists), Einstein's birthplace might be of greater importance than his discovery. Yet the subjective nature of the object of study should not completely discourage the study of it; rather, the difficulty should invite better methods to capture that information.

Besides, to the authors a knowledge graph as it stands is limited, and some notion of weights in relationships must exist in order to better represent knowledge itself.

1.2 Importance-augmented Knowledge Graph

Formally a weighted knowledge graph can be defined as:

$$\mathcal{WKG} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}, f_w\}$$

where $\mathcal{E}, \mathcal{R}$ and $\mathcal{T}$ remain as in $\mathcal{KG}$, and $f_w : \mathcal{T} \to [0, 1]$ is a function from KG triple to the associated weight.

A general method for incorporating weights for link prediction has been studied[1], yet most weights are confidence scores, i.e. measuring the uncertainty of the triple. We see confidence scores as helpful but ultimately uninteresting. Importance as weights are interesting but, as mentioned, perhaps difficult to study due to its subjective nature.

1.3 KG-BERT[2]

In recent years, we have witnessed a meteoric rise of pre-trained language models (PLMs) like BERT[3]. Trained on a very large corpus, PLMs have demonstrated ability to answer questions that require domain knowledge, leading many to hypothesize PLMs to have implicit representation of knowledge. Contrast this with knowledge graphs: PLM is a rich, implicit representation of knowledge whereas KG is a structured, explicit representation of knowledge.

Several attempts have been made to combine the advantages of these two representations. Most notable attempt is KG-BERT, which performs several KG-specific tasks like link prediction, relation prediction, and triple classification using PLMs. Key idea is to convert knowledge graph triples into sequence of tokens, using the sequence as input tto the BERT model. KG-BERT has seen reasonable success, with some improvements made in PKGC[4].

1.4 Importance in LLMs

In the previous section we mentioned how PLM has a rich representation of knowledge. What is meant by a "rich" representation of knowledge? Let us consider a conversation with GPT-4[5], a recent LLM, shown in Figure 1.

We see in the conversation that GPT-4 attempts to order facts about Einstein based on importance. It also notes Einstein's discovery is more important than his birthplace. We can empirically note that GPT-4 can "weight" different relationships based on a notion of importance, whereas KGs are structurally incapable of such task. Thus it is plausible that if we want to create importance-augmented knowledge graphs, we can extract implicit knowledge in them for importance in triples.

While no systematic inquiry has been made as to whether LLMs possess a robust implicit representation of importance of facts, we hypothesize that the success of LLMs imply a rich implicit representation, which serves as the motivation for our experiments.

1.5 Contributions

Therefore, we'd like to leverage PLMs in order to augment KGs with importance. In order to use PLMs, we must fine-tune it to our needs; unfortunately, there exist no publicly available knowledge graph datasets that label edges with importance. Thus we use Wikipedia articles as proxy for importance in a triple: specifically, given triple (h, r, t), position of t in the Wikipedia article of h is used to approximate $f_w(h, r, t)$. Then a BERT model is fine-tuned on this data in order to approximate a general f_w function.

In this paper, we present a model that is capable of computing importance scores using Wikipedia as proxy for importance. This model is then used to augment an existing KG with several important scores for completeness.

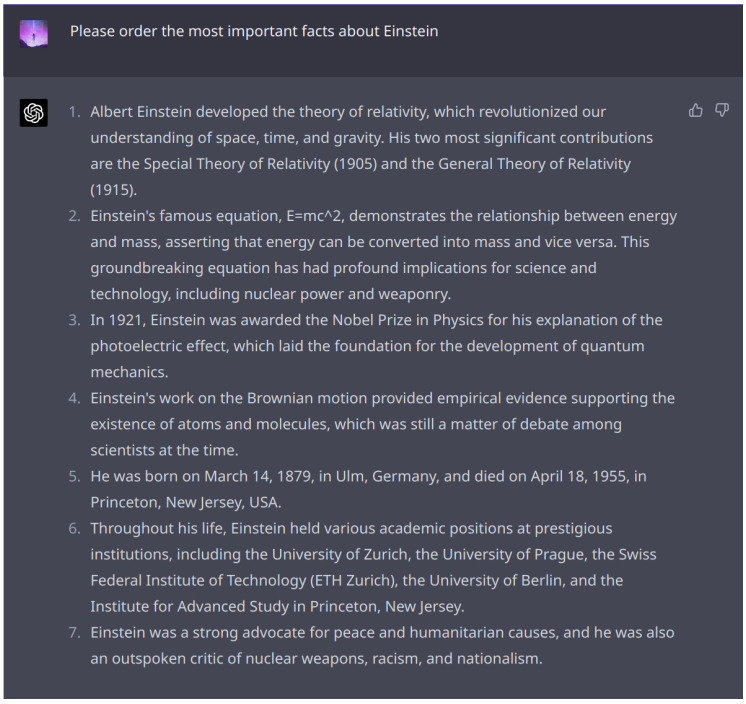

Figure 1: Conversation with GPT-4 on Ordering Einstein Facts based on Importance

2 Importance Score Extraction from Wikipedia

In this section, we will discuss the process of preprocessing Wikipedia articles and the method used to compute importance scores for the generated triples.

2.1 Motivation

Given a triple (h, r, t), we wanted a dataset that gives us $w_{h,r,t}$ where $w_{h,r,t} = f_w(h, r, t)$. Yet as mentioned above, there are no publicly available datasets, so we have to generate $w_{h,r,t}$. Three sources that implicitly capture these relationships came to mind: LLMs, Google, and Wikipedia.

2.1.1 LLMs as Source

As demonstrated above with GPT-4, with proper prompting, we can extract information about importance of facts from LLMs. We however moved away from this approach for several reasons:

1. We prefer human-generated data as the baseline for training our model.
2. Using data generated by LLM to fine-tune BERT seems unwise, and may exacerbate the biases.
3. LLMs are computationally expensive to run inference, and this is an issue when there are triples on the order of a million. A simple backhand calculation: if we generate 1 token per second and each triple has 10 tokens on average, then generating data for 1M triples takes 10M seconds, which is near 4 months.

2.1.2 Google as Source

Let $T_{h,r,t}$ be the textual representation of triple (h, r, t) e.g.

$$T_{\text{Albert\_Einstein,wasBornIn,Germany}} = \text{"Albert Einstein was born in Germany"}$$

Then the number of Google search results of $T_{h,r,t}$ can be used to compute $w_{h,r,t}$, with appropriate normalization. We see this approach as interesting, albeit with high variance and noise. We suggest the number of Google search results be used as a supplementary feature in future work.

2.1.3 Wikipedia as Source

When using Wikipedia as a source, we assume that earlier a fact shows in a Wikipedia article, more important the fact is. We see this assumption as being largely true, as Wikipedia articles are edited by humans with implicit goal of presenting relevant information. The assumption fails sometimes when, for instance, basic information (like birthplace) about a person is presented earlier than more critical information.

Given a triple, we use the index of tail in the Wikipedia article of head in order to compute the importance score. There are different ways to index the tail, which we will go over in subsequent sections.

2.2 Dataset

We used YAGO3-10 (Yet Another Great Ontology 3-10) as the dataset. YAGO3-10 has 123,182 entities, 37 relations, and 1,179,040 triples (of which 1,079,040 we use for training). All entities are identifiers for Wikipedia articles. In order to remove variance in character count, we default to English Wikipedia. Because of this limitation and other issues like website access and inability to find the entity in a Wikipedia article, we could not use near 69.39% of the data for training. Ultimately we have 330,248 triples, or 30.61% of the original dataset for training.

2.3 Preprocessing Wikipedia Articles

Given triple (h, r, t), let

1. $I_{h,t}$ be the link index of entity t in the Wikipedia article of h. Link index here is the index of the entity in the ordered list of all entities in the article, in the order that they appear. We use the term link since named entities in Wikipedia articles link to the corresponding Wikipedia article.
2. $I'_{h,t}$ be the link index of entity t in the Wikipedia article of h, where infobox is ignored. Example of Infobox is shown in Figure 2. Considering links in infobox can be a double-edged sword as sometimes useful entities are in infobox and not in the main article, but also infobox has a regular ordering that does not prioritize importance of facts.
3. $C_{h,t}$ be the character index of the first character of entity t in the Wikipedia article of h. Character index here is the number of characters in the article before the first character of entity.

2.3.1 Limitations

Note how this method does not take the relation type into account. Even though the relation type is an important matter, we assumed that given head h and tail t, in most instances, there is one type of relation between h and t. The mention of t in an article of h will imply t is being mentioned in the context of that one type of relation. Thus the index of t is sufficient to determine the position of the "triple".

2.3.2 Power Decay

Given an index, we use a power decay function to normalize to a real-value between 0 and 1:

$$s(x) = \alpha^x \tag{1}$$

where $s(x)$ is the importance score, x is the position of the tail entity in the article, and α is a positive constant. The power decay function is chosen because it assigns higher importance to entities mentioned earlier in the article while still considering entities that appear later. This function reflects the intuition that earlier-mentioned entities are generally more important, but it allows for the possibility that later-mentioned entities can still be relevant. For $I_{h,t}$ and $I'_{h,t}$, α of 0.95 is chosen. For $C_{h,t}$, α of 0.999 is chosen. The decay rates are chosen based on the magnitude of these values; note that $C_{h,t}$ will be much larger than the other two indices.

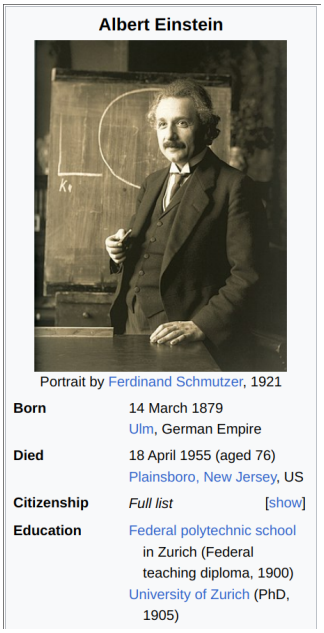

Figure 2: Wikipedia Infobox on Albert Einstein

	$I_{h,t}$	$I'_{h,t}$	$C_{h,t}$
BERT Train Loss (Last Epoch)	0.028	0.070	0.062
BERT Test Loss	0.030	0.061	0.058
RoBERTa Train Loss (Last Epoch)	0.032	0.074	0.066
RoBERTa Test Loss	0.031	0.059	0.062

Table 1: Results for BERT and RoBERTa

3 Experiments

The goal is to train a BERT model in order to fully complete a knowledge graph with importance weights. As mentioned above, 69.39% of the data for YAGO3-10 training dataset do not have weights attached to them. Thus for proper imputation of data, we need to train an appropriate model.

3.1 Methods

Given $h, r, t \in \mathcal{E}$ and $I \in \{I_{h,t}, I'_{h,t}, C_{h,t}\}$, the method is as follows:

1. Convert (h, r, t) to a sequence of tokens, using `relation2text.txt` for r.
2. Let the sequence of tokens be the input to BERT sequence classification model, with the output as a single label I.
3. Use MSE as the loss function and AdamW optimizer for training.

In order to aid training, we preprocessed the Wikipedia data so that $I \in [0, 1]$. 1,079,040 triples were used in the train dataset, and 50,000 triples were used for dev and test datasets each. We used BERT and RoBERTa[6] as the BERT models, learning rate of 1E-5, and 3 epochs.

3.2 Results

We see the results laid out in Table 1. Notably, even though $I_{h,t}$ and $I'_{h,t}$ used the same decay rate, we see that the PLMs are much better able to learn $I_{h,t}$. The reason for this is because $I_{h,t}$ includes the infobox, which leads to much more regular patterns in the importance scores of certain triples.

There are several limitations to the result:

1. It is difficult to tell how good these losses are without having human-labeled data to validate our model. We are assuming that the Wikipedia positions are a good proxy, but when evaluating the results, it is difficult to tell.
2. Wikipedia has many relationships, whereas YAGO3-10 is a subset of these relationships. The low discrepancy between train and test losses evidence low possibility of overfitting, but this may be due to the limitation of the dataset.
3. BERT and RoBERTa do not have significant differences in loss. We expect with a problem as difficult as determining importance of facts, the size of the model will have an effect. Yet there is no difference, which may suggest that these models are opting for a simpler strategy that does not encompass actually learning the task.

Despite these limitations, however, we show that pre-trained language models can learn non-trivial relationships between the "importance" and the triple.

3.3 Importance-augmented Knowledge Graph

In the code that we have provided, we have provided a means of generating importance scores from a knowledge graph. We do this by

1. Given triple (h, r, t), if t can be found within article of h, use the appropriate index for computing importance score.
2. If no t can be found, then use the model we trained in order to generate the importance score, thereby imputing the data.

Thus the result of imputation is a six-column tsv file with head, relation, tail, $I_{h,t}$, $I'_{h,t}$, and $C_{h,t}$. We hope that this dataset can act as a rich augmentation of knowledge graphs enabled by PLMs.

4 Suggested Future Work

4.1 Google as Source

As mentioned above, we suggest using Google as an alternative way to enrichen a knowledge graph. Once controlled for its variance and erraticity, we believe this can be a useful feature to add to a knowledge graph.

4.2 Application to KG-BERT

Initially we sought to use the new representation in order to aid training KG-BERT. We hypothesize that a richer representation of knowledge graphs can help with tasks like link prediction. The computational complexity of these models, however, is very high (taking weeks, up to a month on V100), which is infeasible given allotted time and resources.

5 Conclusion

In this paper, we explored the problem of assigning importance scores to triples in a knowledge graph. We proposed a novel approach by leveraging Wikipedia articles and preprocessing techniques to extract importance scores. Our assumption was that the position of a fact within a Wikipedia article is indicative of its importance. We employed BERT and RoBERTa models to predict importance scores based on the extracted data.

While our approach has shown promise, there is room for improvement. Future work could include using the number of Google search results as a supplementary feature, taking the relation type into account, and exploring different decay functions for normalizing the importance scores. Additionally, the performance of the models can be further enhanced by experimenting with other pre-trained language models, more sophisticated techniques for text processing, and refining the hyperparameters.

In conclusion, our approach demonstrates a viable way to extract and predict importance scores for knowledge graph triples using large language models and Wikipedia articles. This work paves the way for refining and expanding knowledge graphs, enriching explicit knowledge representations.

References

[1] Mojtaba Nayyeri, Gökce Müge Cil, Sahar Vahdati, Francesco Osborne, Andrey Kravchenko, Simone Angioni, Angelo Salatino, Diego Reforgiato Recupero, Enrico Motta, and Jens Lehmann. Link prediction of weighted triples for knowledge graph completion within the scholarly domain. *IEEE Access*, 9:116002–116014, 2021.

[2] Liang Yao, Chengsheng Mao, and Yuan Luo. Kg-bert: Bert for knowledge graph completion, 2019.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[4] Xin Lv, Yankai Lin, Yixin Cao, Lei Hou, Juanzi Li, Zhiyuan Liu, Peng Li, and Jie Zhou. Do pre-trained models benefit knowledge graph completion? a reliable evaluation and a reasonable approach. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3570–3581, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[5] OpenAI. Gpt-4 technical report, 2023.

[6] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.