Language Model-Knowledge Graph Integration

Aisha Khatun
a2khatun@uwaterloo.ca
University of Waterloo
Waterloo, Ontario, Canada

Xiangru Jian
x2jian@uwaterloo.ca
University of Waterloo
Waterloo, Ontario, Canada

ABSTRACT

Recent developments in large language modeling have greatly accelerated the performances of NLP applications. Yet they remain largely dependent on their training data and thus prone to being factually inaccurate and socially biased. It is hard to correct the models after the fact due to their large size requiring high compute and large amounts of supervised training data. This paper proposes a minimal compute, no-pretrain framework for improving language model factual accuracy by incorporating knowledge graph information. Unlike human-written text, facts in knowledge graphs like Wikidata are accurate and free from bias. Comparison with baselines shows that our methods have promise in making language models factually accurate as well as retaining language understanding. We also build a facts dataset to test our work using template sentences and Wikidata entities to further evaluate the proposed system. Our work and dataset are freely available in github.com/tanny411/KG-LM-Integration.

1 INTRODUCTION

The current landscape of NLP is teeming with Large Langauge Models (LLM). Particularly, transformer [40] based models with millions and billions of parameters trained on large amounts of unlabeled text data. The aim was initially to make the models understand language through pretraining so as to facilitate downstream tasks such as classification, translation, named-entity recognition (NER), and so on [8, 12, 36, 48]. This setup would require the pretrained models to be fine-tuned with a given downstream task. Recent developments have focused more on few-shot and zero-shot methods. In this setup, the language models would be able to perform downstream tasks given only a few examples (or no examples) in the prompt. This alleviates the cost, compute, and data requirement of fine-tuning extremely large models and makes these pretrained models usable directly out of box.

The pretrained models are getting larger (by number of parameters) and are trained on larger and larger corpuses with the aim of multitasking [4, 9, 10, 14, 15, 35, 45, 47]. Scaling the models indeed provides gains on downstream tasks [18, 38] and allows for new abilities to be explored [46]. But it is important to remember that LLMs are probabilistic learners. They learn language and reasoning from the text they were trained on and thus their reliability cannot be ensured. For example, LLMs may produce biased [26], privacy-leaking [13], false [22], or outdated information [6, 11]. Using these models in production, especially in fully automated pipelines, can cause serious harm. Updating the large models requires identifying its limitations and re-training with significant amount of supervised data to combat the said limitations. Not only is it expensive, but it may also require fine-grained data and long time to train. To ease model updates and to make them more factually accurate we propose a no-pretrain Knowledge Graph (KG) integrated language model.

Simply stated, knowledge graphs are a bank of facts represented in the form of a graph. Each entity in this graph is connected to other entities through relevant edges. Each fact is a set of three items called `Subject`, `Predicate`, and `Object` forming a `Triple`. For example, (`Ottawa CapitalOf Canada`) is a triple representing a fact. The entities `Ottawa` (Subject) and `Canada` (Object) are connected by an edge `CapitalOf` (Predicate) that describes the relationship between them. These graphs can hold intricately detailed information about entities and the relationships among them. There are several general [20, 41] and domain-specific large knowledge graphs [7] available on the web. Among them, Wikidata [41] is a freely available, constantly updating, and ever-growing knowledge graph. The facts in Wikidata can not only help provide up-to-date factually accurate information to LLMs, but also perform de-biasing by enforcing certain facts. For example, we can assume if a model ingests facts such as (`Man SubclassOf Human`) and (`Woman SubclassOf Human`), it would not be biased against Men or Women for any profession or personality.

The purpose of this paper is to start exploring the area of LLM and KG integration. Specifically, we create a no-pretrain framework where the facts from KG are injected into the LLMs such that the information in LLMs are accurate and bias-free, and the LLMs continue to perform well on the intended downstream tasks. Although some works have attempted to combine LLM with KG, they either work on a small subset of KG [17], attempt to improve only one downstream task [32], or perform full pretraining [34] or fine-tuning [1]. We propose and compare a few integration modules to improve the factual accuracy of LLMs with minimal training while retaining the usability of LLMs as multi-task models. Besides, to test our methods, we create a Facts dataset using template sentences and Wikidata entries (collected with SPARQL queries). Evaluation on Linked Wikitext-2 and Facts dataset shows promise in our approach. This opens the possibility to update and correcting LLMs in a cheap and easy manner.

The rest of the paper is structured as follows: Section 2 contains related works in LLM and KG integration research. Section 3 contains description of datasets used and created for training and evaluation. Section 4 contains details of the architecture proposed and the training methods. Section 5 discusses evaluation results on the proposed method as well as some baselines. Lastly, Section 6 mentions the future direction of this research, and Section 7 summarizes our work and contribution.

2 RELATED WORK

There are some works that combine language models (LM) and knowledge graphs (KG). Most of them are aimed at improving knowledge graphs themselves, such as link prediction [19, 49],

Topic	Capital
Templates	{capital} is the capital of {country}, The capital of {country} is {capital}

SPARQL	SELECT ?capital ?capitalLabel ?country ?countryLabel
	WHERE
	{
	?capital wdt:P1376 ?country.
	?country wdt:P31 wd:Q6256.
	?capital wdt:P31 wd:Q5119.
	SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en". }
	}

Query Results

Capital	CapitalLabel	Country	CountryLabel
wd:Q727	Amsterdam	wd:Q29999	Kingdom of the Netherlands
wd:Q1354	Dhaka	wd:Q902	Bangladesh
wd:Q1362	Islamabad	wd:Q843	Pakistan
wd:Q1519	Abu Dhabi	wd:Q878	United Arab Emirates

Sentences	Amsterdam is the capital of Kingdom of the Netherlands, The capital of Kingdom of the Netherlands is Amsterdam Dhaka is the capital of Bangladesh, The capital of Bangladesh is Dhaka Islamabad is the capital of Pakistan, The capital of Pakistan is Islamabad Abu Dhabi is the capital of United Arab Emirates, The capital of United Arab Emirates is Abu Dhabi
Sentence Selection	Amsterdam is the capital of Kingdom of the Netherlands Dhaka is the capital of Bangladesh The capital of Pakistan is Islamabad Abu Dhabi is the capital of United Arab Emirates
Random Masking	[MASK] is the capital of Kingdom of the Netherlands Dhaka is the capital of [MASK] The capital of [MASK] is Islamabad Abu Dhabi is the capital of [MASK] [MASK] [MASK]

Table 1: Example generation of facts dataset from template sentences and Wikidata queries

triple classification [49, 51], and relation prediction [49]. KG-BERT [49] improves knowledge graph completion tasks by incorporating textual descriptions of entities and relations to the input with the help of BERT [8] and computing scoring functions for triples. Ma et al. [27] attempt to perform KG completion tasks by considering triples as sentences and running them through RNN and CNN. They initialize the embedding matrix with TransE KG embedding [5] and then have it modified during training. Nayyeri et al. [31] go a step forward and combine 4 sources of embeddings: knowledge graph, word, sentence, and document embeddings to perform link prediction in knowledge graphs. The knowledge graph embedding is obtained from the entity while the text-based embeddings are obtained from the textual descriptions of entities. Pretrain-KGE learns KG embedding using sentence embedding of entity definition from BERT and performs link prediction with the usual negative sampling contrastive learning methods [50].

Other works add information from KG into LM in various ways to enhance LM performance. K-BERT trains BERT and an additional KG module to merge these spaces [23]. K-Adapters suggests placing a few adapter layers in between the layers of already pretrained frozen models to add KG information to them [42]. ERICA introduces contrastive loss and two new pretraining tasks but requires significant training to merge the embedding spaces [34]. Google's KELM converts triples into Sentences using a language model itself, specifically T5. The sentences are then fed into a large language model to provide the required information to the model [1]. BERT-MK uses a subgraph of KG (medical) with a pretrained BERT [17]. It adds an aggregator with a triple-learner to incorporate KG information. KELM uses pretrained BERT but trains KG embedding and uses GNN to combine the two spaces [25]. KEPLER jointly optimizes the textual embedding from a pretrained model and KG embedding and performs well on both NLP and KG tasks [43]. It uses MLM (Masked Language Modeling) objective for text and use textual embedding to train convectional KG embedding. KnowBERT performs joint end-to-end training for both LM and KG objectives in a multitask setting [33]. Logan et al. [24] build an architecture where facts relevant to the text are selected from the KG and perform joint training.

Another set of works uses KG and LM to perform specific tasks. Ostendorff et al. [32] perform document classification using Pytorch BigGraph KG embeddings, a large-scale embedding of the entire Wikidata. It concatenates BERT embedding with KG embedding for each book author and runs it through a few MLP (Multi Layer Percepteron) layers to perform the final classification task. THU-ERNIE adds a pretraining task to incorporate KG information [39]. Academic paper search is improved by combining paragraph vectors from the text of a paper and entity embedding from bibliographic knowledge graph to look for similar papers [28]. Mao and Fung [29] measure the semantic relatedness between Unified Medical Language System (UMLS) concepts by combining the space of KG embedding and Word embedding through concatenation. Wang et al. [44] align the word and knowledge graph embedding spaces using Wikipedia-Wikidata Anchors. That is, through the pages from Wikipedia that are connected to Wikidata entities. Santini et al.

[37] perform author name disambiguation of authors of scientific papers. They use LiteralE [21] embedding model to learn the KG embedding and concatenate it with the textual embedding of paper titles generated from BERT [8].

In general, most of the previous studies mentioned above have some major issues. They either require doing the pretraining of the language model again or have a very limited amount of feasible downstream tasks. The former hurts the efficiency badly since pretraining of a general language model is extremely expensive while the latter basically defeats the objective of modern large language models - multitasking. To address the issue, a multi-task language model which integrates information from knowledge graphs into pretrained language embedding needs to be proposed.

3 DATASET

3.1 Linked Wikitext-2 dataset

Linked Wikitext-2[1] [24] is a dataset that connects Wikipedia text spans to Wikidata entities. It contains approximately the same articles as in Wikitext-2 [30] and links to entities that exist in Wikidata [41]. It uses human annotated links between Wikipedia articles to accomplish this. Whenever a span of text in a article is associated with another Wikipedia article, the Wikidata entry corresponding to the latter article is connected to that span of text. Then entity-linkers and coreference resolution tools are used to gather rest of the occurrences of the same Wikidata entry in the text. Table 2 provides some details about the dataset from [24].

	Train	Dev	Test
Documents	600	60	60
Tokens	2,019,195	207,982	236,062
Mention Tokens	207,803	21,226	24,441
Mention Spans	122,983	12,214	15,007
Unique Entities	41,058	5,415	5,625
Unique Relations	1,291	484	504

Table 2: Linked WikiText-2 Corpus Statistics [24].

3.2 Wikidata Facts dataset

Besides Linked Wikitext-2, to test the proposed methods for factual accuracy, we create a dataset of facts from Wikidata. Following the methods of [24], we select 5 types of facts to test: capital, city, birthplace, spouse, writing. We create template sentences for each of these topics. Then we query Wikidata using its SPARQL API[2] to gather entities that are then used to populate actual sentences from the templates. An example of the entire workflow is shown in Table 1 and Table 3 lists the templates used.

Topic	Templates
Capital	{capital} is the capital of {country} The capital of {country} is {capital}
City	{city} is a city of {country} {country} has a city called {city}
Birthplace	{person} was born in {birthplace} {birthplace} is the birthplace of {person}
Spouse	{person1} is the spouse of {person2} {person2} is the spouse of {person1} {person1} is married to {person2} {person2} is married to {person1}
Writing	{person} authored the writing {book} {person} authored {book} {book} is written by {person}

Table 3: Templates used for facts dataset generation

4 METHODOLOGY

This research aims to find an efficient framework to integrate structured information from large universal knowledge graphs into general language models. To achieve this, a knowledge-integrated language representation model (KIG-Bert) is proposed, in which the information of entities and relations contained in certain input sentences will be integrated into their embedding given by language models. KIG-Bert, as shown in Figure 1, consists of three modules, i.e. KG-based module, LM-based module and Integration module[3]. For an input sentence, KG-based module produces an embedding of it based on the pretrained embedding of a given knowledge graph (i.e. Wikidata in this study), while the LM-based module outputs the embedding from a pretrained language model, like Bert and its variants. After the two parallel modules, we design the integration module to integrate the KG-based embedding of the input into the LM-based one to generate a new embedding that considers the information from both the knowledge graph as well the language model.

4.1 KG-based module

KG-based module consists of three technical components, which are the entity extractor, the pretrained embedding of the chosen knowledge graph, and a customized encoder. For an input sentence, the entity extractor finds out all the entities in it which are also in the given knowledge graph and then looks up their embeddings in the set of embeddings of that knowledge graph. After that, we apply the customized encoder to transform those entities' embedding into the sentence's embedding, denoted as KB-based embedding.

In this study, we use Linked Wikitext-2, which already contains text span to KG entity linkage. Therefore, a separate entity extractor design was not necessary. Besides, we choose to build our model on the knowledge graph in Wikidata. Therefore, we can directly adopt the above-mentioned large-scale embedding of the entire Wikidata, called Pytorch BigGraph KG embeddings, as the pretrained embedding used in this study. This set of embeddings is produced by the well-known TransE model and has provably good performance in a large range of related studies.

We give the strict definition that a token corresponds to an entity (and the other way around) if the label of that entity literally contains the token. For example, the entity "howler monkey" corresponds to two tokens, namely "howler" and "monkey". The

[1] https://rloganiv.github.io/linked-wikitext-2
[2] https://query.wikidata.org/

[3] Related code of this study can be found at https://github.com/tanny411/KG-LM-Integration

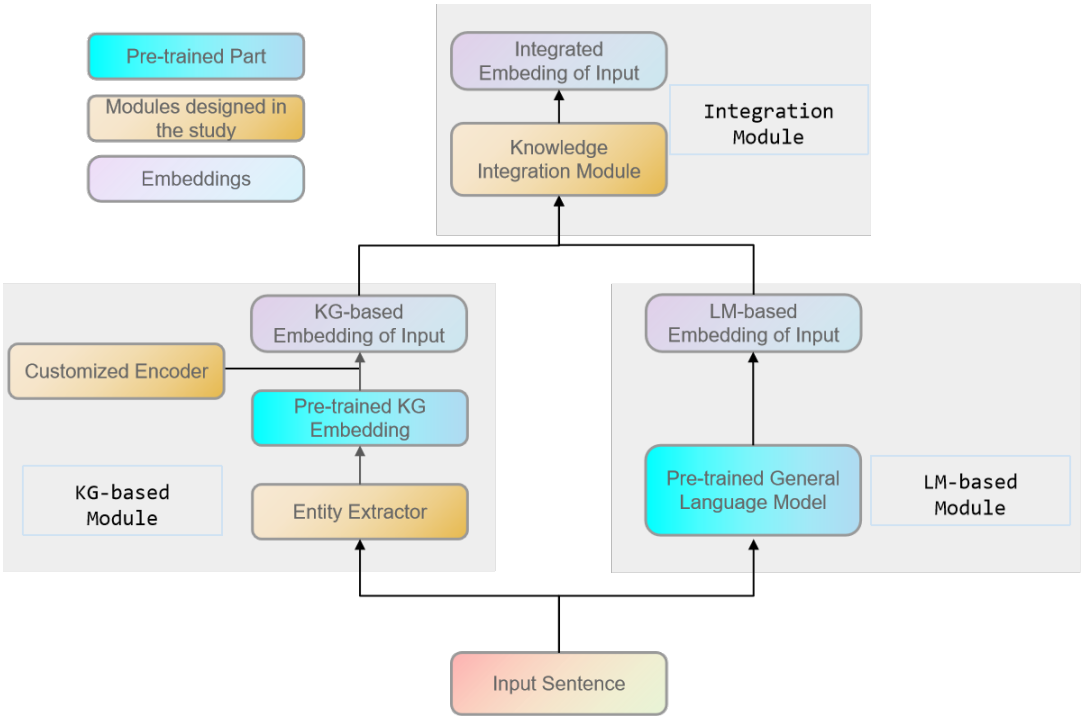

Figure 1: Architecture of KIG-Bert

customized encoder designed in this study follows an intuitive heuristic that encodes every token in the input sentence with the embedding of its corresponding entity in the knowledge graph if it has one. Then, we assign an all-zero embedding to any other tokens that do not correspond to any entity. Note that if multiple tokens together form an entity, we consider all these tokens corresponding to this entity. Therefore, the KG-based embeddings are just the embedding of corresponding entities. In this way, we can assign KG-based embedding for every token in the input sentence.

4.2 LM-based module

LM-based module is essentially the pretrained language model, which, in this study, is one of the lightweight variants of Bert, called DistilBERT [36]. DistilBERT is a transformers model, smaller and faster than BERT, which was pretrained on the same corpus as Bert but in a self-supervised fashion, using the BERT base model as a teacher. Since it is already pretrained, it takes an input sentence and directly outputs its embedding, denoted as LM-based embedding.

4.3 Integration module (KIM)

With both KG- and LM-based embedding, the most critical work in the integration module is to integrate them together to obtain a more informative embedding of the input sentence, which is the final output of KIG-Bert. The design of this knowledge integration module (KIM) will be the key point of this study. To the best knowledge of the authors, there is no previous study specifically on this task. However, some related studies have been done on embedding alignment and integration between different language models or knowledge graphs. For the alignment between different language embedding, Artetxe et al. [2] introduces mapping matrix to align the embedding of a pair of words that share similar meanings from two different languages. Fu et al. [16] utilize bidirectional GAN to find the non-linear transformation functions between different spaces of sentences representations. Aside from language embedding, Baumgartner et al.[3] propose FedCoder, as a variant of autoencoder to integrate multiple knowledge graph embedding spaces.

In this study, we adopt a similar method of concatenation and transformation described in [37], the details are presented in Figure 2. Specifically, we project the LM- and KG-based embeddings of the input sentence into two feed-forward layers respectively and obtain output embeddings that have the same dimension, represented by the red and blue tensor in Figure 2. The two projected embeddings are concatenated together and fed to the encoder consisting of several feed-forward layers. The output of the encoder will be the integrated embedding (ItgtEmb), as detailed in Equation 1.

$$\text{ItgtEmb} = \text{Encoder}\{\text{Concat}[\text{FFLM}(\text{LM-Emb}), \text{FFKG}(\text{KG-Emb})]\} \tag{1}$$

The major reason why we choose this design instead of attention-based methods is that we need the integration module to be as lightweight as possible. Otherwise, the integration of embeddings becomes meaningless since we can always add the pretraining task to the original language model and redo its pretraining. But now, with much fewer layers and parameters, the training of the integration module can be fast and efficient. Also, we find that our integration module with an extra layer of self-attention just

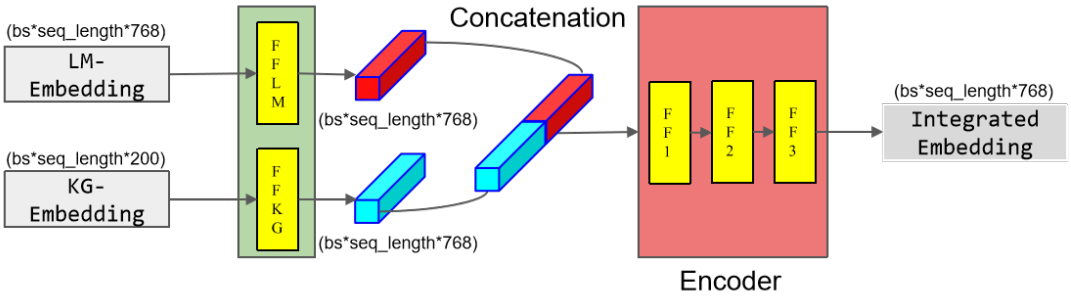

Figure 2: Architecture of the Integration module

before the output performs at least no better in evaluation, which we present in Section 5.4.

4.4 Training Objective

Finally, we need to design a training objective to train the integration module. Since the module is similar to any general pretrained language model producing embeddings of tokens in a sentence, the parameters in KIM will be trained in a self-supervised manner using the tasks usually utilized by other language models, like Bert. To make the module understand the semantics of natural language, Masked Language Modeling (MLM) objective is used just like Bert and its variants. The module should also be capable of interpreting the information of entities within each input sentence and producing its embedding accordingly. Therefore, we add an entity prediction task that uses the integrated embedding of each token in the sentence to predict its corresponding entity in the knowledge graph. Note that we only perform this task on the tokens that really correspond to an entity in the knowledge graph. In fact, this task is very similar to the masked token prediction except that the vocabulary we predict from is one of all the entities in the knowledge graph instead of all the possible tokens in the language model. Therefore, both tasks are multi-class classifications and so we can apply cross-entropy loss to both of them, yielding a total loss as the sum of two losses from two tasks respectively.

5 RESULTS AND ANALYSIS

5.1 Evaluation Dataset

We use the training set of Linked Wikitext-2 dataset to train the integration module. We use Linked Wikitext-2 test set as well as Wikidata facts dataset to evaluate the module. The results from the evaluation are presented in Section 5.4. Note that KG-based module and LM-based module are not re-trained. Therefore, the evaluation in the section reflects the performance of the Knowledge Integration Module (KIM).

5.2 Metrics

KIM module is trained with two simultaneous training objectives: MLM corresponding to LM, and entity detection corresponding to KG. We intend to improve LM with the factual accuracy of KG. Therefore, while training we need to make sure that the model retains both LM and KG capabilities. With this end in view, we formulate and report two metrics: the accuracy of LM token detection called LM-Accuracy, and the accuracy of KG entity detection called KG-Accuracy. These metrics are defined by Equation 2 and 3.

$$\text{LM-Acc} = \frac{\text{Number of tokens detected correctly}}{\text{Total Masked Tokens}} \tag{2}$$

$$\text{KG-Acc} = \frac{\text{Number of entities detected correctly}}{\text{Total Masked entity present in Vocabulary}} \tag{3}$$

5.3 Benchmarks

To argue the performance of KIM designed in this study, we need to set some benchmarks for the comparison. There are in general two aspects to judge the module. Firstly, the comparison between the integrated embedding of our model to the two inputs, namely the pretrained natural language embedding from DistilBERT and the entity embedding from Pytorch BigGraph. Secondly, we compare our integrated embedding to that of alternative KIMs with different architectures. In sum, we come up with three suitable benchmarks for the comparison.

5.3.1 LM-Raw. Firstly, to evaluate the quality of integrated embedding from the aspect of natural language, we want to compare the LM-accuracy of proposed KIM to that of the input sentence embedding generated by the Bert model we used in this study, i.e. DistilBERT. If our KIM has no less LM-accuracy than this input embedding, we can argue the KIM does capture the important information in natural language. Note that the classifier used in the calculation of the LM-accuracy of KIM is trained by the training set. Therefore, the only way to perform the evaluation fairly is to train a simple one-layer classifier in a similar way to obtain the LM-accuracy of embeddings generated by DistilBERT instead of directly using pretrained DistilBERT model. The latter one is much stronger than the former one since it's trained on a much larger corpus, which is definitely not a fair game in this study. We denote this benchmark as LM-Raw.

5.3.2 KG-Raw. Similarly, we also design a baseline called KG-Raw as a single-layer classifier over the KG entities. It gives the KG-accuracy of the original pretrained embedding of Pytorch BigGraph. If our KIM is capable of attaining a higher KG-accuracy than this benchmark does, we can argue the KIM also captures the semantics of entity well.

5.3.3 Alt-KIM. In the end, as mentioned at the end of Section 4, we also design some alternative integration modules with more complex structures. Due to the limited time and computational resources, we only include one such module for evaluation. Alt-KIM has an extra layer of self-attention, which is quite representative since several previous studies related to the integration between language model and knowledge graph adopted attention-based models. This module serves as a benchmark to show the quality of our KIM compared to other possible designs.

5.4 Experimental Result

The result of performance on two test sets of the proposed KIM and all the benchmarks are presented in Table 4. By design, the baseline LM-Raw and KG-Raw can not provide KG-accuracy and LM-accuracy, respectively. They are based on one-half of the input without access to the other half.

Table 4: Evaluation results of KIM

Model	Linked Wikitext-2		Wikidata facts	
	LM-Acc(%)	KG-Acc(%)	LM-Acc(%)	KG-Acc(%)
LM-Raw	18.72	N/A	**11.84**	N/A
KG-Raw	N/A	34.11	N/A	73.25
Alt-KIM	31.51	36.54	1.974	88.54
Our KIM	**35.71**	**38.46**	3.947	**89.81**

Firstly, the proposed KIM module has an overall better performance compared to the two baselines built on the input embedding, i.e. KG-Raw and LM-Raw. This indicates that the information in the embeddings of the language model and knowledge graph is captured well in the KIM module. At least through the simple evaluation in this study, one can argue that it is possible to have a more accurate and expressive language model with the help of information contained in the entities of knowledge graphs. Note that there is an exception in the result where LM-Raw has better performance on the dataset of Wikidata facts. Each sentence in that test set is relatively short (10-20 tokens), so the modified language model can hardly learn useful information in the context. More training may help improve the LM abilities of KIM for short sentences.

Besides, the performance of the proposed KIM is better on both metrics compared to that of Alt-KIM. Since the attention calculation is computationally expensive, the complexity of Alt-KIM (as well as other possible attention-based methods) is much larger than that of our KIM. Therefore, our KIM appears to be better in both accuracy and efficiency.

5.5 Qualitative Analysis

The intention of this study is to enhance the performance of large language models with information from knowledge graphs. We can confirm that our model is able to accomplish the desired goals only when information in the knowledge graph can make up for the weakness of classic language models. Although this idea was previously explored through experiments on a variety of applications, our study contributes some new examples.

For some of the facts in the Wikidata facts dataset, like,

1. `The capital of Slovenia is Ljubljana.`
2. `Accra is the capital of Ghana.`

it is relatively hard for classic language models to predict the name `Ljubljana` and `Ghana` among all other tokens since the useful tokens `Ljubljana`, `Ghana`, `Slovenia` and `Accra` rarely appear in the training corpus of those models. However, from the perspective of entity predicting, the problem is much easier in the sense that the embedding of `Slovenia` and `Accra` must be very close to that of `Ljubljana` and `Ghana`, respectively. This is because the task here is similar to the object prediction performed in multiple knowledge graph embedding methods, including TransE. The experimental results also support this analysis since the integrated embedding yield accurate predictions in entity prediction while failing to include the correct LM token in the top 10 predictions in mask token prediction.

In sum, the motivation of the study can be tested and validated by the experiments presented in this paper. Therefore, more rigorous research can be performed following this study.

6 FUTURE WORK

There are many challenges we need to address to find a better method and pipeline for the integration of language models and knowledge graphs.

First, we need more data for training and testing so that the KIM module can have better generalization ability to all kinds of natural language corpus. To use unlabeled text, an extractor is required that can find entities and relations from plain text efficiently and correspond them to their entity ID accurately. This would involve entity detection, disambiguation, and coreference resolution. Second, alternative architectural designs of modules need to be tested to serve as further benchmarks and help perform a rigorous ablation study of the designed KIM module. The optimal architecture will be such that it is capable of merging LM and KG spaces effectively while being lightweight, therefore not requiring too much training. Lastly, the proposed modules need to be tested on NLP benchmark tasks to assess their LM abilities and on real-world factual accuracy and debiasing tasks to determine their effectiveness on a wide variety of knowledge types.

7 CONCLUSION

Language models are probabilistic learners that perform really well on a variety of NLP tasks including text generation, classification, and question-answering. But recent studies have revealed the unreliable nature of language models when it comes to factual accuracy and social biases. Various studies have attempted to solve this issue by mostly de-biasing the training data or training the models with supervised data of relevant facts. Given the computation and time required to re-train such large models, we propose a no-pretrain method of incorporating facts into existing language models. Our method involves using pretrained embeddings of knowledge graph entities and training a small integrator module to combine both LM and KG embedding. Our results show promise in this direction. Both KG and LM evaluation tasks perform better than a plain single-layer models. We also present a facts dataset to asses the factual accuracy of language models. Extensive architecture search, evaluation on factual accuracy datasets, and comparison with LM baselines are topics of further studies.

REFERENCES

[1] Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2020. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. *arXiv preprint arXiv:2010.12688* (2020).

[2] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, 451–462. https://doi.org/10.18653/v1/P17-1042

[3] Matthias Baumgartner, Daniele Dell'Aglio, Heiko Paulheim, and Abraham Bernstein. 2023. Towards the Web of Embeddings: Integrating multiple knowledge graph embedding spaces with FedCoder. *Journal of Web Semantics* 75 (2023), 100741. https://doi.org/10.1016/j.websem.2022.100741

[4] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. *GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.* https://doi.org/10.5281/zenodo.5297715 If you use this software, please cite it using these metadata..

[5] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems*, C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf

[6] Wenhu Chen, Xinyi Wang, and William Yang Wang. 2021. A Dataset for Answering Time-Sensitive Questions. https://doi.org/10.48550/ARXIV.2108.06314

[7] The UniProt Consortium. 2020. UniProt: the universal protein knowledgebase in 2021. *Nucleic Acids Research* 49, D1 (11 2020), D480–D489. https://doi.org/10.1093/nar/gkaa1100 arXiv:https://academic.oup.com/nar/article-pdf/49/D1/D480/35364103/gkaa1100.pdf

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

[9] Brown et al. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 1877–1901. https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[10] Chowdhery et al. 2022. PaLM: Scaling Language Modeling with Pathways. https://doi.org/10.48550/ARXIV.2204.02311

[11] Kasai et al. 2022. RealTime QA: What's the Answer Right Now? https://doi.org/10.48550/ARXIV.2207.13332

[12] Liu et al. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. https://doi.org/10.48550/ARXIV.1907.11692

[13] Nicholas Carlini et al. 2021. In *USENIX Security Symposium*. 2633–2650. https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting

[14] Ouyang et al. 2022. Training language models to follow instructions with human feedback. https://doi.org/10.48550/ARXIV.2203.02155

[15] Zhang et al. 2022. OPT: Open Pre-trained Transformer Language Models. https://doi.org/10.48550/ARXIV.2205.01068

[16] Zuohui Fu, Yikun Xian, Shijie Geng, Yingqiang Ge, Yuting Wang, Xin Dong, Guang Wang, and Gerard de Melo. 2020. ABSent: Cross-Lingual Sentence Representation Mapping with Bidirectional GANs. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 05 (Apr. 2020), 7756–7763. https://doi.org/10.1609/aaai.v34i05.6279

[17] Bin He, Di Zhou, Jinghui Xiao, Qun Liu, Nicholas Jing Yuan, Tong Xu, et al. 2019. Integrating graph contextualized knowledge into pre-trained language models. *arXiv preprint arXiv:1912.00147* (2019).

[18] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling Laws for Neural Language Models. https://doi.org/10.48550/ARXIV.2001.08361

[19] Knowledge Graph Embedding 2022. *Knowledge Graph Embedding.* Retrieved Sep 27, 2022 from https://en.wikipedia.org/wiki/Knowledge_graph_embedding

[20] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, S. Auer, and Christian Bizer. 2015. DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6 (2015), 167–195.

[21] Mingyang Li, Neng Gao, Chenyang Tu, Jia Peng, and Min Li. 2021. Incorporating Attributes Semantics into Knowledge Graph Embeddings. In *2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. 620–625. https://doi.org/10.1109/CSCWD49262.2021.9437876

[22] Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. TruthfulQA: Measuring How Models Mimic Human Falsehoods. https://doi.org/10.48550/ARXIV.2109.07958

[23] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 2901–2908.

[24] Robert L. Logan, IV, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. 2019. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Florence, Italy.

[25] Yinquan Lu, Haonan Lu, Guirong Fu, and Qun Liu. 2021. KELM: knowledge enhanced pre-trained language representations with message passing on hierarchical relational graphs. *arXiv preprint arXiv:2109.04223* (2021).

[26] Li Lucy and David Bamman. 2021. Gender and Representation Bias in GPT-3 Generated Stories. In *Proceedings of the Third Workshop on Narrative Understanding*. Association for Computational Linguistics, Virtual, 48–55. https://doi.org/10.18653/v1/2021.nuse-1.5

[27] Minbo Ma, Fei Teng, Wen Zhong, and Zheng MA. 2019. A Sentence-RCNN embedding model for Knowledge Graph Completion. In *2019 IEEE 14th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*. 484–490. https://doi.org/10.1109/ISKE47853.2019.9170315

[28] Gengchen Mai, Krzysztof Janowicz, and Bo Yan. 2018. Combining Text Embedding and Knowledge Graph Embedding Techniques for Academic Search Engines. In *Semdeep/NLIWoD@ISWC*.

[29] Yuqing Mao and Kin Wah Fung. 2020. Use of word and graph embedding to measure semantic relatedness between Unified Medical Language System concepts. *Journal of the American Medical Informatics Association : JAMIA* 27 (2020), 1538 – 1546.

[30] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer Sentinel Mixture Models. *ArXiv* abs/1609.07843 (2016).

[31] Mojtaba Nayyeri, Zihao Wang, Mst. Mahfuja Akter, Mirza Mohtashim Alam, Md Rashad Al Hasan Rony, Jens Lehmann, and Steffen Staab. 2022. Integrating Knowledge Graph embedding and pretrained Language Models in Hypercomplex Spaces. https://doi.org/10.48550/ARXIV.2208.02743

[32] Malte Ostendorff, Peter Bourgonje, Maria Berger, Julian Moreno-Schneider, Georg Rehm, and Bela Gipp. 2019. Enriching bert with knowledge graph embeddings for document classification. *arXiv preprint arXiv:1909.08402* (2019).

[33] Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge Enhanced Contextual Word Representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 43–54. https://doi.org/10.18653/v1/D19-1005

[34] Yujia Qin, Yankai Lin, Ryuichi Takanobu, Zhiyuan Liu, Peng Li, Heng Ji, Minlie Huang, Maosong Sun, and Jie Zhou. 2020. ERICA: improving entity and relation understanding for pre-trained language models via contrastive learning. *arXiv preprint arXiv:2012.15022* (2020).

[35] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. https://doi.org/10.48550/ARXIV.1910.10683

[36] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. https://doi.org/10.48550/ARXIV.1910.01108

[37] Cristian Santini, Genet Asefa Gesese, Silvio Peroni, Aldo Gangemi, Harald Sack, and Mehwish Alam. 2022. A Knowledge Graph Embeddings Based Approach for Author Name Disambiguation Using Literals. *Scientometrics* 127, 8 (aug 2022), 4887–4912. https://doi.org/10.1007/s11192-022-04426-2

[38] Aarohi Srivastava and Rastogi et al. 2022. Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models. https://doi.org/10.48550/ARXIV.2206.04615

[39] THU-ERNIE 2022. *THU-ERNIE.* Retrieved Sep 27, 2022 from https://paddlepedia-readthedocs-io.translate.goog/en/latest/tutorials/pretrain_model/THU-ERNIE.html?_x_tr_sl=zh-CN&_x_tr_tl=en&_x_tr_hl=en&_x_tr_pto=sc

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. https://arxiv.org/pdf/1706.03762.pdf

[41] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. *Commun. ACM* 57, 10 (sep 2014), 78–85. https://doi.org/10.1145/2629489

[42] Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808* (2020).

[43] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation. *Transactions of the Association for Computational Linguistics* 9 (03 2021), 176–194. https://doi.org/10.1162/tacl_a_00360 arXiv:https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00360/1923927/tacl_a_00360.pdf

[44] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph and Text Jointly Embedding. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1591–1601. https://doi.org/10.3115/v1/D14-1167

[45] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned Language Models

are Zero-Shot Learners. In *International Conference on Learning Representations*. https://openreview.net/forum?id=gEZrGCozdqR

[46] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682* (2022).

[47] BigScience et al Workshop. 2022. BLOOM: A 176B-Parameter Open-Access Multilingual Language Model. https://doi.org/10.48550/ARXIV.2211.05100

[48] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. *XLNet: Generalized Autoregressive Pretraining for Language Understanding*. Curran Associates Inc., Red Hook, NY, USA.

[49] Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for Knowledge Graph Completion. *ArXiv* abs/1909.03193 (2019).

[50] Zhiyuan Zhang, Xiaoqian Liu, Yi Zhang, Qi Su, Xu Sun, and Bin He. 2020. Pretrain-KGE: Learning Knowledge Representation from Pretrained Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics, Online, 259–266. https://doi.org/10.18653/v1/2020.findings-emnlp.25

[51] Xiaohan Zou. 2020. A survey on application of knowledge graph. In *Journal of Physics: Conference Series*, Vol. 1487. IOP Publishing, 012016.