MEM-KGC: Masked Entity Model for Knowledge Graph Completion With Pre-Trained Language Model

BONGGEUN CHOI1, DAESIK JANG2, AND YOUNGJOONG KO2

1Department of Electrical and Computer Engineering, Sungkyunkwan University, Suwon, Gyeonggi-do 16419, South Korea
2Department of Computer Science and Engineering, Sungkyunkwan University, Suwon, Gyeonggi-do 16419, South Korea

Corresponding author: Youngjoong Ko (youngjoong.ko@gmail.com)

This work was supported by the Institute for Information & Communications Technology Planning & Evaluation (IITP) through Korea Government [Ministry of Science and ICT (MSIT)], Development of Knowledge Evolutionary WiseQA Platform Technology for Human Knowledge Augmented Services under Grant 2013-2-0013.

ABSTRACT The knowledge graph completion (KGC) task aims to predict missing links in knowledge graphs. Recently, several KGC models based on translational distance or semantic matching methods have been proposed and have achieved meaningful results. However, existing models have a significant shortcoming—they cannot train entity embedding when an entity does not appear in the training phase. As a result, such models use randomly initialized embeddings for entities that are unseen in the training phase and cause a critical decrease in performance during the test phase. To solve this problem, we propose a new approach that performs KGC task by utilizing the masked language model (MLM) that is used for a pre-trained language model. Given a triple \((h, r, t)\), we mask the tail entity and consider the head entity and the relation as a context for the tail entity. The model then predicts the masked entity from among all entities. Then, the task is conducted by the same process as an MLM, which predicts a masked token with a given context of tokens. Our experimental results show that the proposed model achieves significantly improved performances when unseen entities appear during the test phase and achieves state-of-the-art performance on the WN18RR dataset.

INDEX TERMS Knowledge graph completion, link prediction, masked language model, pre-trained language model.

I. INTRODUCTION

Knowledge graphs (KGs) are collections of real-world knowledge represented as a triple in the form of \((h, r, t)\), denoting head entity, relation and tail entity, respectively. In recent years, KGs have significantly contributed to improvements in various knowledge-driven applications, such as question answering [1] [2] [3], information retrieval [4] and dialogue systems [5]. Despite the increasing use of KGs, they usually suffer from incompleteness because of several missing links as they have been manually built. To complete KGs by finding missing links, the knowledge graph completion (KGC) task (a.k.a link prediction) has attracted growing interest.

There are two main approaches to conduct KGC task. A typical approach for KGC task is the knowledge graph embedding (KGE) method, which represents entities and relations as embedding vectors and calculates the scores of triples using those embeddings. Although models using KGE method have achieved meaningful results, there is a fatal drawback that they can only learn entity embeddings when entities appear during the training phase [6]. If an entity does not appear at the training phase, the entity embedding remains randomly initialized and untrained. We call this the unseen entity problem. As a result, a triple containing an unseen entity has almost no possibility to be the top-ranked triple because an entity with the highest score among all entities is chosen as a proper one for a given input, “head entity and relation” or “tail entity and relation”. In addition, because the models using KGE method infer missing links by only using entity and relation embeddings, there is no way to utilize lexical
information of entities or relations (i.e., descriptions or definitions), which can enrich the embeddings.

On the other hand, Yao et al. [7] proposed KG-BERT that learns a triple as a sequence of text to contain its lexical information sufficiently. KG-BERT not only can process unseen entities but also can make more plausible embeddings using descriptions or definitions of entities and relations even when an unseen entity appears at the test phase. Furthermore, Kim et al. [8] integrated KGC with the relation prediction and relevance ranking tasks following the framework of the multi-task learning method [9] to obtain better performance among high-scored entities. Although these methods based on pre-trained language models have shown higher scores for proper entities on average, they still show considerably lower performances on Hits@1 and Hits@3 than the models using KGE method do. Moreover, pre-trained language models usually have a high time complexity, so the inference time is extremely slow.

To leverage the effectiveness of KGE method and pre-trained language models, we propose the masked entity model for KGC (MEM-KGC) that enhances KGC task using the process of the masked language model (MLM). As we make the model predict a proper entity, which is similar to the process of MLM, the MEM-KGC extremely reduces inference time while enriching embeddings by using a pre-trained language model. One of the important features affecting the performances of existing models is negative samples [10]. Unlike these existing models, because our model does not need to train negative samples, it can minimize the effect of sampling methods and reduce the training time significantly.

Considering the prediction process of the proposed model, our model needs to make more plausible prediction for a proper entity with a given context, head entity and relation by using only one [MASK] token embedding, which is used for link prediction of many entities. Thus, we additionally train the model to predict not only entities but also their super-classes with given entity descriptions; we call these tasks as the entity prediction (EP) and the super-class prediction (SP), respectively.

Our major contributions can be summarized as follows:
1) In this study, we propose a new approach to effectively conduct KGC task by utilizing a typical MLM. The proposed model achieved better performances than existing models based on pre-trained language models, such as KG-BERT and the multi-task learning for knowledge graph completion (MTL-KGC) [8] while extremely reducing the inference time.
2) The proposed model significantly improves performances comparing to existing models using KGE method through the multi-task approach of the EP and SP, particularly when unseen entities appear at the test phase.
3) MEM-KGC does not need to train negative samples. In particular, this condition does not make the model seriously depend on what negative samples are chosen, and it requires very short training time.
4) Finally, we constructed and will open new corpora with the super-class information of two benchmark datasets, namely WN18RR-Sup and FB15k-237-Sup.

II. RELATED WORK
A. KNOWLEDGE GRAPH EMBEDDING MODELS
Models using KGE method can be categorized into two groups based on how to calculate the scores of triples: translational distance models and semantic matching models [11]. The former group, which includes TransE [12] and RotatE [13], learns distance-based scoring functions considering relations as the translational operation between entities. Conversely, the latter group, including DistMult [14], learns the scoring functions based on the similarity between entities and relations. During the test phase, such models infer scores across all entities with a given head and relation (or relation and tail) and takes an entity resulting in the highest score as proper one.

In translational distance models, a relation usually performs a translation between two entities in a triple, and then a distance between the two entities is used as a score of the triple. The most representative translational distance model, TransE, represents entities and relations with the same dimension of embedding vectors [12]. Given a triple \((h, r, t)\), the scoring function of TransE is defined as \(f_r(h, t) = -\|h + r - t\|\), supposing that entities and relations satisfy \(h + r \approx t\), where \(h, r, t \in \mathbb{R}^d\). However, TransE does not perform well on 1-N, N-1 and N-N relations [15]. Thus, RotatE maps entities and relations to complex embeddings, following Euler’s identity, and defines the scoring function as \(f_r(h, t) = -r^T (h \circ r - t)\), where \(h, r, t \in \mathbb{C}^d, |r| = 1\) and \(\circ\) denotes Hadamard (or element-wise) product [13]. Each element of a relation embedding performs a rotation between two entities in a complex space. This feature makes RotatE well infer relational patterns: symmetry/antisymmetry, inversion and composition. Furthermore, HAKE maps entities into polar coordinate system for modeling the semantic hierarchy [16]. DualE represents relations as compositions of translation and rotation operation by employing dual quaternion space [17].

Meanwhile, DistMult is a semantic matching model that calculates scores based on the similarity between two entities and a relation embedding of a triple [18]. DistMult defines the scoring function as \(f_r(h, t) = h^T \text{diag}(r)t\), where \(h, r, t \in \mathbb{R}^d\). Although DistMult is good at capturing the compositional semantics of relations via a matrix multiplication, this simplified model is usually less expressive and not strong enough for KGs [16].

B. PRE-TRAINED LANGUAGE MODELS FOR KGC
Recent research has shown that pre-trained language models, such as the bidirectional encoder representations from transformers (BERT), significantly improve performances on several natural language processing tasks [19]. Yao et al. [7] first utilized BERT for KGC task and named it
as KG-BERT. This approach treats a triple as a sequence and turns KGC into a sequence classification problem with the binary cross-entropy loss function. KG-BERT performs well on average when inferring valid triples but is far behind the state-of-the-art performances in evaluating higher ranks, such as Hits@1 and Hits@3. To improve high-rank performance, Kim et al. [8] combined two additional tasks, i.e., relation prediction and relevance ranking, with KGC task through the multi-task learning framework of the multi-task deep neural networks (MT-DNN) [9]. MTL-KGC shows the effectiveness of multi-task learning and improves performances on Hits@1, Hits@3 and MRR as compared with KG-BERT. More recently, Zhang et al. [20] tried to integrate a typical KGE method with a pre-trained language model by initializing entity and relation embeddings from the pre-trained language model. They first encode entities and relations through the pre-trained language model, use the encoded vectors as initial embeddings of entities and relation, and fine-tune the pre-trained language model with loss of a model using KGE method. After the fine-tuning, they extract entity and relation embeddings through the fine-tuned pre-trained language model and use them as initial embeddings. Finally, they train typical models, such as TransE or RotatE, using the embeddings.

III. PROPOSED MODEL

In this section, we introduce the proposed model and its extensions. First, subsection A describes the baseline model and subsections B and C describe its extensions, namely, entity prediction (EP) and super-class prediction (SP).

A. BASELINE MODEL

Originally, KGC models predict a tail entity when given a head entity and relation in the case of tail-batch. Conversely, they predict a head entity when given a relation and tail entity in the case of head-batch. As motivated by MLM, we slightly changed KGC to employ MLM. In the view of the tail entity when the case is tail-batch, a given head entity and relation are a context of a triple. Then KGC task can be adopted to MLM, which predicts a masked token with a given context, using the masked tail entity. We call this modified KGC as the masked entity model (MEM-KGC). With the above process, MEM-KGC predicts what the given masked entity should be. For example, given a triple (iPhone, made_by, Apple), we mask Apple and iPhone alternately, such as (iPhone, made_by, [MASK]) and ([MASK], made_by, Apple); the model then predicts iPhone and Apple by the entity classifier from among all entities in the case of the tail-batch and the head-batch, respectively.

An input for the MEM-KGC is as same as KG-BERT except for [MASK] tokens. For simplicity, we describe the input only in the case of the tail-batch as follows:

\[ x_{\text{base}} = [CLS] h h_{\text{des}} [SEP] r [SEP] [MASK], \]

where \( h \) and \( r \) denote a head entity and relation of a triple, respectively, and \( h_{\text{des}} \) denotes the description of \( h \).

For the prediction, the final hidden vector corresponding to the masked entity, \( E_{\text{mask}} \in \mathbb{R}^H \), where \( H \) is the number of hidden dimensions, is fed into the output softmax over the number of entities, \( K \), as in standard language models:

\[ E_{\text{mask}} = \text{Encoder}(x_{\text{base}})_{\text{mask}}, \quad (2) \]
\[ \hat{y} = \text{softmax}(E_{\text{mask}} \cdot W^T_e), \quad (3) \]

where \( \hat{y} \in \mathbb{R}^K \) denotes the output probability vector for a prediction for which entity the [MASK] token should be predicted among overall entities and \( W_e \in \mathbb{R}^{K \times H} \) denotes a learnable parameter matrix. To train the model, we use the cross-entropy loss function:

\[ L_{\text{base}} = -\frac{1}{K} \sum_{i=1}^{K} y_i \log \hat{y}_i \quad (4) \]

Figure 1 shows the overall process of the baseline model for the case of the head-batch (Figure 1(a)) and tail-batch (Figure 1(b)).

B. ENTITY PREDICTION

One of the fatal limits of KGE method is that it is weak whenever unseen entities appear during the test phase. Because the embeddings of the unseen entities remain as being randomly initialized and untrained when applying KGE method, the scores of the triples containing the unseen entities become also meaningless scores.

To let the model learn the information of such unseen entities, we add EP task so that the model predicts an entity through [MASK] token embedding with a give description of the entity:

\[ x_{\text{ep}} = [CLS] [\text{MASK}] [SEP] e_{\text{des}} [SEP] \quad (5) \]

Because the role of the entity classifier in EP is the same as that of the baseline which is to predict an entity among all entities that is proper to a given context, the model uses the same classifier from the baseline for EP as shown in Figure 2.

We also calculate the loss of EP using the cross-entropy loss function:

\[ E_{\text{mask}} = \text{Encoder}(x_{\text{ep}})_{\text{mask}}, \quad (6) \]
\[ \hat{y} = \text{softmax}(E_{\text{mask}} \cdot W^T_e), \quad (7) \]
\[ L_{\text{ep}} = -\frac{1}{K} \sum_{i=1}^{K} y_i \log \hat{y}_i \quad (8) \]

C. SUPER-CLASS PREDICTION

As considering both of the baseline and EP, we predict entities only using the [MASK] token embedding but most entities are likely to be too specific in KGs. Thus, it makes the model difficult to predict a proper entity for a given context whenever unseen entities appear either as masked or as given. Because the model has to distinguish different entities using the same [MASK] token embedding without learning any information
of them. Thus, we integrated EP with SP following a typical multi-task learning method, as shown in Figure 3. Since there are no other ways to learn the information of unseen entities, the model can learn abstractive information of unseen entities only through SP. For example, the model has to both predict actor, which is one of super-classes, with [CLS] token embedding either when an input is description of Johnny Depp or an input is that of Brad Pitt.

We use a different learnable parameter, $W_s \in \mathbb{R}^{S \times H}$, where $S$ is the number of super-classes, for SP because the role of the classifier is different with that of the baseline and EP. The entity classifier predicts an entity for the [MASK] token, whereas the super-class classifier predicts a super-class of the given entity description with the [CLS] token embedding.

We also use the cross-entropy loss function to calculate the loss of SP as follows:

$$E_{cls} = \text{Encoder}(x^{ep})_{cls},$$  \hfill (9)  

$$\hat{y} = \text{softmax}(E_{cls} \cdot W_s^T),$$  \hfill (10)  

$$L_{sp} = -\frac{1}{S} \sum_{i=1}^{S} y_i \log \hat{y}_i$$  \hfill (11)  

The final loss of EP with SP is the weighted sum of the loss from each task. We experimentally verify the chosen weight, $\lambda$, for SP loss in Section V-B:

$$L_{ep+sp} = L_{ep} + \lambda \cdot L_{sp}$$  \hfill (12)
TABLE 1. Statistics of datasets.

| Dataset  | entities | relations | super-classes | train | validation | test |
|----------|----------|-----------|---------------|-------|------------|------|
| WN18RR   | 40,943   | 11        | 14,371        | 86,835| 3,034      | 3,134|
| FB15k-237| 14,541   | 237       | 1,992         | 272,115| 17,535     | 20,466|

IV. EXPERIMENTS

A. DATASETS AND IMPLEMENTATION DETAILS

1) DATASETS

We used two benchmark datasets, WN18RR [21] and FB15k-237 [22], to evaluate the proposed model because they are the most commonly used datasets for KGC. The WN18RR dataset is a subset of WordNet [23], which consists of lexical data. Entities from WN18RR are words or short phrases having lexical information. There are 40,943 entities and 11 relations from 93,003 triples.

The FB15k-237 dataset is a subset of Freebase [24], a graph-structured knowledge base consisting of general human knowledge, such as names of actors or locations. Entities and relations in FB15k-237 are more specific than those in WordNet. There are 14,541 entities and 237 relations from 310,116 triples. We used the same descriptions as those provided by Yao et al. [7] and Xie et al. [6] for the WN18RR and FB15k-237 entities, respectively. Furthermore, to apply SP, we manually collected the super-classes by using “Synset Hypernym” of the WordNet for the WN18RR and WikiData for the FB15k-237. Finally, we will open new corpora with the collected super-class information, namely WN18RR-Sup and FB15k-237-Sup. Table 1 summarizes the two datasets.

In addition, there are 210 and 28 test triples containing unseen entities which are 6.7% and 0.13% of the total test triples for the WN18RR and FB15k-237, respectively.

2) EXPERIMENTAL SETTINGS

For the training, we used the pre-trained BERT-base for the encoder and fine-tuned over a multi-task setup for 10 epochs on Quadro 8,000 GPU. We used a mini-batch size of 32, a max sequence length of 128 and Adam optimizer [25] with learning rates 5e−5 and 3e−5 for WN18RR and FB15k-237, respectively. The training procedure of MEM-KGC follows the framework of MT-DNN [9]. In each training step, a mini-batch \( b_t \) is selected among the baseline and EP. Then, the model is updated by the loss (e.g., \( L_{base} \) or \( L_{ep+sp} \)) of the selected task \( t \).

For the inference, we follow the filtered method used by Bordes et al. [12] for a fair comparison with prior studies. Let \( \mathcal{E} \) be an entity set and \( \mathcal{T} \) be a set of all triples. Then, given a test triple \( (h, r, t) \), negative triple set, \( \mathcal{N} \), is \( \mathcal{N} = \{(h, r, t')|t' \in \mathcal{E} \land (h, r, t') \notin \mathcal{T}\} \) in the case of the tail batch. Finally, a candidate test set is \( \mathcal{U} = (h, r, t) \cup \mathcal{N} \) for the given test triple \( (h, r, t) \). We call the triple \( (h, r, t) \) as a positive example and \( (h, r, t') \in \mathcal{N} \) as negative examples for the given positive example \( (h, r, t) \). Then a model has to rank the positive example in the highest among them.

B. RESULTS OF THE PROPOSED MODEL

Table 2 demonstrates that the additional two tasks improve performances over the baseline model. Performances increased on both datasets when EP and SP were added. The improvement of the performances is lower on the FB15k-237 dataset than on the WN18RR dataset, because the two tasks, EP and SP, were designed to improve performances on unseen entities. There are only 28 triples containing an unseen entity in the FB15k-237 dataset, which accounts for only 0.13% of the total test triples. Meanwhile, there are 210 test triples containing an unseen entity, which accounts for 6.7% of total test triples in the WN18RR dataset. The different composition ratio of unseen entities among two datasets makes different impact of performance improvement. We specifically reported the effectiveness of the EP and SP for the unseen entities in Section V-A.

C. PERFORMANCE COMPARISON WITH EXISTING MODELS

The comparison results between the proposed model and existing models are shown in Table 3. Comparing to the existing models based on a pre-trained language model, our model significantly improved the performances. Furthermore, the proposed model achieved the state-of-the-art for the WN18RR dataset comparing to the previous state-of-the-art model, improving by 7.5%, 3.7%, 10.4% and 12.6% on the mean reciprocal rank (MRR), Hits@1, Hits@3, and Hits@10 metrics, respectively. Moreover, despite the small number of unseen entities in the FB15k-237 dataset, our model achieved competitive performances to DualE [17] of the state-of-the-art model.

V. ANALYSIS

A. ANALYSIS ON UNSEEN ENTITIES

We have performed the unseen entity analysis on the WN18RR dataset because there are too small amounts of triples containing unseen entities in the FB15k-237 dataset to obtain the reliability of the analysis. To verify that two extended tasks, EP and SP, improve performances when the unseen entities appear during the test phase, our model was separately evaluated in the following two cases.

In the first case, seen entities are the answers, that is to say, the model has to predict the seen entities as answers with the given unseen entities and relations for all of the positive and negative examples, e.g. triple (unseen entity, relation, seen entity). Thus, when considering the test process of existing models, a positive example does not have any penalty to negative examples in this case. Because our model was trained to predict seen entities with a given context during the training phase, it presented better performance both than existing models and one in the second case. Herein, RotatE [13] is employed for comparison as a representative model for KGE method, which cannot process the unseen entities; RotatE shows almost zero performances as shown in Table 4.
In the second case, the model has to predict unseen entities as answers with given seen entities and relations for positive examples. It is a much harder case for both existing models and ours in the first case because all negative examples consist of a triple (seen entity, relation, seen entity). It is almost impossible that the score of a positive example with an unseen entity larger than ones of negative examples. Our baseline model also had never been trained to predict unseen entities while training. In the first case, it is almost impossible that the score of a positive example with an unseen entity larger than ones of negative examples. Our baseline model also had never been trained to predict unseen entities while training. Table 5 indicates that every model shows lower performance than that of the first case. Nevertheless, Table 5 also indicates that our model improves the performance when EP and SP were added. This is because the model was trained to predict unseen entities through EP and learned abstractive information of unseen entities through SP.

In addition, we visualized the logit values of the answer entities (unseen entities) from the second case. As shown in Figure 4, the logit values of the extended models located higher position than ones of baseline in most test examples (x-axis). Figure 5 illustrates the normal distributions of the logit values from Figure 4 and the graphs of the unseen entities gradually approximate to those of the logit values of the case where unseen entities appeared as neither given nor masked during the test phase.

Finally, EP and SP showed a positive effect on the performance improvement both of the above cases.

### B. ANALYSIS ON THE LOSS RATIO FOR THE EXTENDED TASKS

When training the model in the frame of multi-task learning, its performance depends on the combination of the loss ratio. Thus, we attempted to figure out the λ for the golden loss

---

**TABLE 2.** Performance of the proposed model along proposed tasks. The values in bold indicate the best performance.

| Model      | WN18RR          | FB15k-237        |
|------------|-----------------|------------------|
|            | MRR  Hits@1 Hits@3 Hits@10 | MRR  Hits@1 Hits@3 Hits@10 |
| Baseline   | 0.533 0.473 0.570 0.636 | 0.339 0.249 0.372 0.522 |
| w/ EP      | 0.557 0.475 0.604 0.704 | 0.346 0.253 0.381 0.531 |
| w/ EP and SP | 0.572 0.489 0.620 0.723 | 0.349 0.260 0.382 0.524 |

**TABLE 3.** Performance comparison with the existing KGC models. We refer to Kim et al. [8] for the performances of KG-BERT because Yao et al. [7] did not report the performances on Hits@1, Hits@3 and MMR metrics.

| Model | WN18RR          | FB15k-237        |
|-------|-----------------|------------------|
|       | MRR  Hits@1 Hits@3 Hits@10 | MRR  Hits@1 Hits@3 Hits@10 |
| TransE [12] | 0.226  -  -  0.501 | 0.294  -  -  0.465 |
| DistMult [14] | 0.430 0.390 0.440 0.490 | 0.241 0.155 0.263 0.419 |
| RotatE [13] | 0.476 0.428 0.492 0.571 | 0.338 0.241 0.375 0.533 |
| HAKE [16] | 0.497 0.452 0.516 0.582 | 0.346 0.250 0.381 0.524 |
| DualE [17] | 0.492 0.444 0.513 0.584 | 0.365 0.268 0.400 0.559 |
| *KG-BERT [7] | 0.219 0.095 0.243 0.497 | 0.237 0.169 0.260 0.427 |
| MTL-KGC [8] | 0.331 0.203 0.383 0.597 | 0.267 0.172 0.298 0.458 |
| Pretrain-KGE [20] | 0.488 0.437 0.509 0.586 | 0.350 0.250 0.384 0.554 |
| Ours    | **0.572 0.489 0.620 0.723** | **0.349 0.260 0.382 0.524** |

**TABLE 4.** The performance when unseen entities are given.

| Model | WN18RR          |
|-------|-----------------|
|       | MRR  Hits@1 Hits@3 Hits@10 |
| RotatE [13] | 0.005 0.000 0.005 0.010 |
| Baseline | 0.603 0.524 0.652 0.757 |
| w/ EP | 0.623 0.519 0.695 0.819 |
| w/ EP and SP | 0.679 0.586 0.733 0.833 |

**TABLE 5.** The performance when unseen entities are masked.

| Model | WN18RR          |
|-------|-----------------|
|       | MRR  Hits@1 Hits@3 Hits@10 |
| RotatE [13] | 0.000 0.000 0.000 0.000 |
| Baseline | 0.000 0.000 0.000 0.000 |
| w/ EP | 0.056 0.005 0.052 0.157 |
| w/ EP and SP | 0.096 0.043 0.091 0.210 |

**FIGURE 4.** Logit values of the unseen entities as answers.

In addition, we visualized the logit values of the answer entities (unseen entities) from the second case. As shown in Figure 4, the logit values of the extended models located higher position than ones of baseline in most test examples (x-axis). Figure 5 illustrates the normal distributions of the logit values from Figure 4 and the graphs of the unseen entities gradually approximate to those of the logit values of the case where unseen entities appeared as neither given nor masked during the test phase.

Finally, EP and SP showed a positive effect on the performance improvement both of the above cases.
In this study, we have presented a new approach for KGC task. The proposed model, called MEM-KGC, not only well trains lexical information using a pre-trained language model but also infers the outputs within a competitive inference time. Specifically, our model showed significantly improved performance through the multi-task learning method when a larger number of unseen entities appear in the inference phase. Furthermore, we constructed and will open two corpora, the WN18RR-Sup and FB15k-237-Sup datasets.

In future, we will focus on modeling an architecture that can well handle new entities—which are completely new that do not belong to the given entity set $E$—with a competitive time complexity.

### REFERENCES

[1] D. Sorokin and I. Gurevych, “Modeling semantics with gated graph neural networks for knowledge base question answering,” in *Proc. 27th Int. Conf. Comput. Linguistics*, pp. 3306–3317, Santa Fe, NM, USA, Aug. 2018.

[2] N. De Cao, W. Aziz, and I. Titov, “Question answering by reasoning across documents with graph convolutional networks,” in *Proc. Conf. North. Minneapolis, MIN, USA, Jun. 2019, pp. 2306–2317.

[3] X. Huang, J. Zhang, D. Li, and P. Li, “Knowledge graph embedding based question answering,” in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, Jan. 2019, pp. 105–113.

[4] C. Xiong, R. Power, and J. Callan, “Explicit semantic ranking for academic search via knowledge graph embedding,” in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 1271–1279.

[5] S. Moon, P. Shah, A. Kumar, and R. Subba, “OpenDialKG: Explaienable conversational reasoning with attention-based walks over knowledge graphs,” in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 545–554.

[6] R. Xie, Z. Liu, J. Jia, H. Luan, and M. Sun, “Representation learning of knowledge graphs with entity descriptions,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, 2016, pp. 1–7.

[7] L. Yao, C. Mao, and Y. Luo, “KG-BERT: BERT for knowledge graph completion,” 2019, arXiv:1909.03193. [Online]. Available: https://arxiv.org/abs/1909.03193

[8] B. Kim, T. Hong, Y. Ko, and J. Seo, “Multi-task learning for knowledge graph completion with pre-trained language models,” in *Proc. 28th Int. Conf. Comput. Linguistics*, 2020, pp. 1737–1743.

[9] X. Liu, P. He, W. Chen, and J. Gao, “Multi-task deep neural networks for natural language understanding,” in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, Florence, Italy, 2019, pp. 4487–4496.

[10] H. Huang, G. Long, T. Shen, J. Jiang, and C. Zhang, “RatE: Relation-adaptive translating embedding for knowledge graph completion,” in *Proc. AAAI Conf. Comput. Linguistics*, Barcelona, Spain, 2020, pp. 684–690.

[11] Q. Wang, Z. Mao, and B. Wang, “Knowledge graph embedding: A survey of approaches and applications,” *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 12, pp. 2724–2743, Dec. 2017.

[12] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 2787–2795.

[13] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang, “Rotate: Knowledge graph embedding by rotational invariance in complex space,” in *Proc. Int. Conf. Learn. Represent.*, 2019.

[14] B. Yang, Y. Yih, X. He, J. Gao, and L. Deng, “Embedding entities and relations for learning and inference in knowledge bases,” in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015.

[15] Z. Wang, J. Zhang, J. Feng, and Z. Chen, “Knowledge graph embedding by translating on hyperplanes,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 28, 2014, pp. 1112–1119.

[16] Z. Zhang, J. Cai, Y. Zhang, and J. Wang, “Learning hierarchy-aware knowledge graph embeddings for link prediction,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 3065–3072.

[17] Z. Cao, Q. Xu, Z. Yang, X. Cao, and Q. Huang, “Dual quaternion knowledge graph embeddings,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 8, May 2021, pp. 6894–6902.

[18] B. Yang, Y. Yih, X. He, J. Gao, and L. Deng, “Embedding entities and relations for learning and inference in knowledge bases,” 2014, arXiv:1412.6575. [Online]. Available: https://arxiv.org/abs/1412.6575

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, Hum. Lang. Technol., Minneapolis, MIN, USA, vol. 1, Jun. 2019, pp. 4171–4186.

### C. ANALYSIS ON THE TIME COMPLEXITY

One of the advantages of MEM-KGC is fast training and inference time when compared to existing models. Although the proposed model uses a pre-trained language model, it does not train negative samples during training phase. In addition, MEM-KGC does not need to calculate scores for all entities one by one because it infers scores of all entities from the classifier at once in the test phase. In this way, we significantly reduced the training and inference time as shown in Table 7.

We used Tesla V100 GPU to estimate the training time and inference time per an example.

### VI. CONCLUSION AND FUTURE WORK

In this study, we have presented a new approach for KGC task. The proposed model, called MEM-KGC, not only well trains lexical information using a pre-trained language model but also infers the outputs within a competitive inference time.
B. Choi et al.: MEM-KGC With Pre-Trained Language Model

[20] Z. Zhang, X. Liu, Y. Zhang, Q. Su, X. Sun, and B. He, “Pretrain-KGE: Learning knowledge representation from pretrained language models,” in Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP), 2020, pp. 259–266.

[21] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, “Convolutional 2D knowledge graph embeddings,” in Proc. AAAI Conf. Artif. Intell., vol. 32, 2018, pp. 1811–1818.

[22] K. Toutanova and D. Chen, “Observed versus latent features for knowledge base and text inference,” in Proc. 3rd Workshop Continuous Vector Space Models Compositionality, 2015, pp. 57–66.

[23] G. A. Miller, “WordNet: A lexical database for English,” Commun. ACM, vol. 38, no. 11, pp. 39–41, 1995.

[24] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, “Freebase: A collaboratively created graph database for structuring human knowledge,” in Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD), 2008, pp. 1247–1250.

[25] P. D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. 3rd Int. Conf. Learn. Represent. (ICLR), San Diego, CA, USA, May 2015, pp. 1–15.

BONGGEUN CHOI is currently pursuing the M.S. degree with the Department of Electrical and Computer Engineering, Sungkyunkwan University, Suwon, South Korea. His research interests include natural language processing, graph neural networks, knowledge graph completion, and knowledge base questions answering.

DAESIK JANG is currently pursuing the M.S. degree with the Department of Computer Science and Engineering, Sungkyunkwan University, Suwon, South Korea. His research interests include natural language processing, knowledge graph completion, and commonsense question answering.

YOUNGJOONG KO received the Ph.D. degree from the Department of Computer Science, Sogang University, Seoul, South Korea, in 2003. He worked at LG-EDS, from 1996 to 1997. Since 2004, he has been joined the faculty of Dong—A University, Busan, where he leads the Intelligent System Laboratory, Department of Computer Engineering. He worked with the Computational Linguistics and Information Processing Laboratory (CLIP), University of Maryland, College Park, as a Visiting Scholar, from 2011 to 2012. In 2019, he moved Sungkyunkwan University, Suwon. His research interests include natural language processing, machine learning (deep neural networks), spoken dialogue systems, question answering systems, knowledge graph engineering, information retrieval, and big data analysis.

* * *