Improving Entity Disambiguation using Knowledge Graph Regularization

Zhi-Rui Tam[0000−0001−9968−2416], Yi-Lun Wu, and Hong-Han Shuai*[0000−0003−2216−077X]

National Yang Ming Chiao Tung University
{ray.eed08g, w86763777.eed08g, hhshuai}@nctu.edu.tw

Abstract. Entity disambiguation plays the role on bridging between words of interest from an input text document and unique entities in a target Knowledge Base (KB). In this study, to address the challenges of global entity disambiguation, we proposed Conditional Masked Entity Model using Knowledge Graph Regularization (CMEM-KG), based on a conditional masked language model, in which multiple mentions in a context can be disambiguated in one forward pass. In addition, to address the long-tailed distribution of global entity disambiguation, we proposed a link prediction regularization, in which the entity embeddings were jointly learned to predict knowledge graph links to prevent the model from overfitting. Compared to other global entity disambiguation models, the model proposed in this study exhibited improved performance on six public datasets without an iterative decoding.

Keywords: Entity Disambiguation · Parallel Decoding

1 Introduction

Entity disambiguation, which is also known as entity linking, refers to the task of assigning entities (such as famous locations, brands, or companies) mentioned in a sentence to a unique identity in a given knowledge base. For example, "A jaguar is cruising on the highway" and "A jaguar is hiding in the jungle" are both mentions of "jaguar", but they refer to two different entity labels based on the contexts. The former refers to a `Jaguar_Car`, whereas the latter refers to a `Jaguar_Animal`. Entity disambiguation is an essential task across multiple natural language processing (NLP) applications, such as coreference resolution [24], canonicalization [8] and question answering [17].

Generally, entity disambiguation can be classified into local entity disambiguation and global entity disambiguation. Local entity disambiguation [14, 16] focuses on disambiguating mentions based on their local contexts. However, the locality limits the performance when the information provided in a local context is not sufficient. In contrast, global entity disambiguation [4, 11, 26] overcomes the limitation of the local entity disambiguation by maximizing the coherence between

* Corresponding Author.

entities and the context of a document. However, despite the progress in global entity disambiguation, the increase in computing complexity with an increase in the number of mentions in a given document has limited its applications.[1]

Although tremendous efforts have been devoted to overcome the challenges of global entity disambiguation, three issues have restricted the effective application of this task. 1) **Topic Coherence**: When a given context is with multiple mentions, global entity disambiguation should choose entities that belongs to the same topic as the document. For example, "Clark Kent is Superman cover up identity" and the "Superman" mention is known to link to the comic character, global entity disambiguation model should choose the fictional character in the comic for the mention of "Clark_Kent" rather than the music producer since the former fits under the same topic as "Superman". 2) **Slow Decoding Speed**: To achieve an optimal coherence between multiple mentions, entity disambiguation models must explore all possible entity combinations. Existing state-of-the-art methods [15, 32] adopt iterative methods, of which the computation complexity increases linearly with an increase in the number of mentions. Although iterative method is better than the exponential complexity from previous works [23, 25], a more efficient entity disambiguation is still desirable. 3) **Long-tailed Distribution.** The distribution of entities usually follows a long-tailed distribution. A dataset is majorly composed of the tail of a distribution. Thus, many entities are only contained by a small number of labeled documents, which cannot be disambiguated well owing to overfitting by the model. Although this issue can be alleviated by filtering out infrequent entities, this approach significantly reduces the amount of entity vocabulary; thus, limiting the number of entity predictions.

To address these challenges, in this study we proposed a novel framework, namely Conditional Masked Entity Model using Knowledge Graph Regularization (CMEM-KG). Particularly, to address the topic coherence challenge, a bi-directional transformer-based decoder was used to model the coherence between mentions. The decoder can implicitly learn to determine the coherent entity via a self-attention between mentions. To address the slow decoding speed, we formulated an entity disambiguation task as a conditional entity disambiguation problem, which is similar to the masked language model [7], to facilitate the parallel prediction of all the unknown entities. To overcome the long-tailed distribution, CMEM-KG utilized a knowledge graph link prediction as a regularization term to improve the disambiguation of long-tailed entities. As a frequently-mentioned entity may share a common relation with less-frequently mentioned entities, updating the parameters of frequently-mentioned entities can also update those of the less-frequent entities as they are linked through a common relation. In contrast to the previous approaches [21, 28], the method proposed in the study learned knowledge graph links separately. Furthermore, we proposed an entropy-based filtering method to trim excessive nodes in a knowledge graph. The entropy-based filtering method implicitly represented the number of edge degrees and link probability in one value; thus, simplifying the hyperparameters

[1] Previous work [9] proves the exact solution for the coherence of a global entity is considered as an NP-hard problem.

used to remove unwanted nodes. The performance of the model on six public datasets was investigated and the results revealed that the performance of the proposed approach on a CoNLL testing dataset without finetuning was at least 24% better than those of all baseline models. Our implementation is available at: https://github.com/basiclab/CMED.

2 Related Work

Neural network-based entity disambiguation achieves a state-of-the-art performance [12, 16, 15] owing to the utilization of multiple training objectives to model the relation between an entity and its surrounding context (i.e., words and entities within the same documents). However, this approach relies on finding the co-occurrence matrix between entity–entity and entity–words. Consequently, the application of this approach is limited owing to the sparse matrix. To address this issue, additional features such as entity types [32] and Wikipedia link information, are often required to further improve the performance of this approach [9, 30].

Recently, pre-trained transformer-based models (such as BERT [7]) have attracted significant attention as a promising alternative for entity disambiguation. [2] reported that the use of pre-trained bi-directional transformers, such as BERT, can easily achieve competitive scores. In addition, [15] modified the BERT model by iteratively disambiguating one entity with the highest confidence score and feeding the entity back into the model as inputs to disambiguate the next entity until all entities are resolved. However, the use of the same model to encode both context and entities exhibits a sub-optimal performance as they follow two different distributions. In contrast, the model proposed in this study utilized two modules to encode context and entity separately.

As knowledge graphs store interlinked descriptions of entities, recent studies have incorporated knowledge graph into entity disambiguation [21, 28]. For example, [21] utilized a local knowledge graph triplets as part of the input sequence. However, this approach does not solve the long-tailed problem as the knowledge graph relations are still part of the model inputs, which falls under the same long-tailed distribution. [28] concatenated the knowledge graph embedding of the entity candidate with the mention embedding as the auxiliary information. Nevertheless, this approach only tackles local entity disambiguation, while our model focuses on global entity disambiguation, which is more challenging.

3 Conditional Masked Entity Model using Knowledge Graph Regularization (CMEM-KG)

3.1 Architecture Overview

Given a set of M mentions $m_1, m_2, m_3, ..., m_M$ found in a context C and a vocabulary of V predefined entities $e_1, e_2, ..., e_V$, the entity prediction task aims to predict which entity should be linked to each of the mentions. To tackle this problem, we proposed Conditional Masked Entity Model using Knowledge Graph

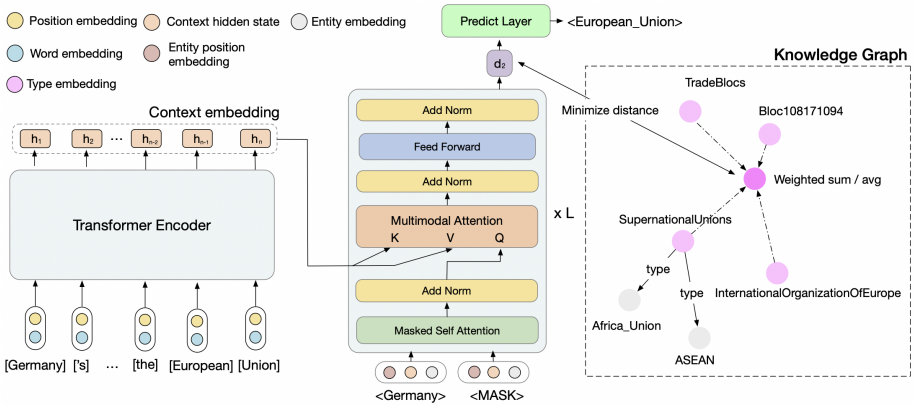

Fig. 1. The model architecture of the proposed CMEM-KG. The decoder predicts the missing entity marked by [MASK].

Regularization (CMEM-KG). The architecture of CMEM-KG consistsed of a text encoder and bidirectional Transformer-based [29] entity decoder. First, we masked out the entity linked for each mention. Thus, the model must predict the masked entity conditioned to the context features and mention features. Figure 1 shows the overall architecture of the model proposed in this study. The text encoder encodes the mentioned context into a contextual representation, which consisted of position embeddings and word token embeddings, similar to that of the Transformer. Thereafter, the entity decoder utilized the hidden states, h, of the mention tokens and entity embedding (denoted as: $E \in R^{V_e \times H}$), and outputs a set of hidden states d representing the mention features. If the mention tokens span over more than one-word tokens, the average of the hidden states is utilized instead. Next, we passed the mention features to a linear model to predict the entity of the mention. It is worth noting that the entity decoder attends to both left and right entities because the optimal order of entity disambiguation was unknown. Moreover, similar to the sequence-to-sequence transformer model, the entity decoder subjects the outputs from the text encoder to a cross self-attention.

3.2 Entity Prediction

After deriving the mention representation (d), the entity linked to the mention was predicted. Based on the study of [10], which utilized a parallel decoding model, we used a conditional masked loss by randomly replacing half of the entity labels with a mask token $[MASK]$ after which the probability strategy proposed by [10] was utilized. The decoder aims to predict the correct entity label for every masked token conditioned to the mention embedding and context representation. The prediction of each entity label can be learned parallelly, as the probability that an entity is masked is independent to other entities.

To predict the original entity of the masked token, the decoder output of the i-th mention, denoted by $d_i \in R^H$, was multiplied by the full entity embedding

matrix E before the softmax function was applied:

$$y_{entity-i} = softmax(E \cdot d_i). \tag{1}$$

To ensure that both spaces were aligned in the same entity space without any bias shift, we did not include the commonly-used bias term in the prediction model of the masked language model.

3.3 Link Prediction Regularization

Although the proposed model could predict entities based on context and mention features, the prediction still suffers from overfitting the entities with the small number of mentions in the Wikipedia paragraphs because neural networks can easily memorize the mentions linked to an entity without learning the surrounding contexts. This indicates that the performance of the model for unseen mentions within similar contexts may be poor. Therefore, in this study, we proposed a novel knowledge graph regularization to prevent overfitting by the model.

Particularly, the knowledge graph enables the linking of a commonly-used entity to a rare entity as a triplet relation (i.e., (e_i, r, e_j)), where r is the relation between e_i and e_j. For example, `Jaguar_Car` (commonly-used) can be linked to `William_Lyons` (rare) through a `foundedBy` relation. Entities in a triplet relation share the same entity vocabulary found in the entity disambiguation task. Moreover, the knowledge graph also contains types (*e.g.*, `Jaguar_Car` and `William_Lyons` have the types of `Car` and `AutomotivePioneers`, respectively), the type of the entity can be extracted as type sets $t_1, t_2 ..., t_K$ with a total vocabulary size K. Accordingly, each entity was linked to a various subsets of types to represent the characteristics of an entity. To leverage the information from the knowledge graph, one possible approach is to use the graph embedding [22, 3] and margin distance loss similar to TransE [1], which is able to learn *one-to-one* relations of (e_i, r_{ij}, t_j). Let ϕ_{e_i}, ϕ_{t_j}, and $\phi_{r_{ij}}$ respectively represent the embeddings of the i-th entity, the j-th type, and the relation between them. Moreover, $(\hat{e_i}, r_{ij}, \hat{t_j})$ represents the corrupted entity triplet pair, i.e., either the entity or type is replaced by a random entity or type. The triplet loss minimizes the distance for (e_i, r_{ij}, t_j) and maximize the distance for $(\hat{e_i}, r_{ij}, \hat{t_j})$, and can be calculated as follows.

$$L_{triplet} = \gamma + d(\phi_{e_i} + \phi_r, \phi_{t_j}) - d(\phi_{\hat{e}_i} + \phi_r, \phi_{\hat{t}_j}), \tag{2}$$

where γ is the margin hyperparameter. However, the relation between entity to types is a one-to-many mapping (e_i, r, T_k), *i.e.*, an entity e_k has a set of valid type $T_k = \{t_{k,1}, t_{k,2}, \cdots, t_{k,N_k}\}$ with size N_k. According to a previous study [19], this approach cannot learn one-to-many mapping results. Therefore, in this study, the set of types was initially passed through an aggregation function f, after which model the output embedding was modelled as one-to-one relations. Let r_{type} denote the relation between entity and type. Thus, the loss functions of links[2] and types were be represented using:

[2] Only entity to entity relations are minimized in link loss.

$$\begin{aligned} L_{type}(e_i, T_i) &= \gamma + d(\phi_{e_i} + \phi_{r_{type}}, f(T_i)) \\ &\quad - d(\phi_{\hat{e}_i} + \phi_{r_{type}}, f(\hat{T}_i), \\ L_{link} &= \frac{1}{N}\sum L_{triplet} + \frac{1}{N}\sum L_{type}. \end{aligned} \tag{3}$$

The first approach of f is to use the average sum of all the type embeddings for aggregation:

$$f_{avg}(T_k) = \frac{1}{N}\sum_{i=1}^{N}(\phi_{T_{ki}}). \tag{4}$$

However, the average function may not be optimal because not all types share equal contribution to form the entity representation. Therefore, we proposed an attention-weighted sum, in which the query vector was replaced with the representation of an entity, *i.e.*,

$$\begin{aligned} Q = F_Q(\phi_t), K = F_K(e) \\ \alpha = softmax(\frac{Q^T K}{\sqrt{d}}) \\ f_{attn}(T_k) = \sum_{i}^{T} \alpha_i \ \phi_{T_{k,i}}, \end{aligned} \tag{5}$$

Using the attention-weighted sum in Eq. 5, the model dynamically assigned a weight value α to each type. Subsequently, the normalized weights were computed by calculating the dot product of the entity representation entity and the type embeddings, which were scaled using the dimension of entity representation $\sqrt{d}$ after which the normalized weights were passed through the softmax function. The normalized weights were used to calculate the weighted contribution corresponding to each of their type embeddings.

The final objective function of CMEM-KG was:

$$-\sum y \ log(p(x)) + \lambda_1 L_{link} + \lambda_2 \sum L_{type}(d_i, r_{type}, T_i), \tag{6}$$

where the first term is the cross entropy loss for entity disambiguation, λ_1 and λ_2 are the weighted terms for the regularization term, and $d \in \mathbb{R}^h$ is the output embedding from the decoder. The key idea here is that the type relations were uniformly sampled; thus, ensuring that each entity embedding was uniformly updated to alleviate the problem caused by the long-tailed distribution.

3.4 Entropy-based Type Filtering

During the inference stage of entity disambiguation, the relation and type from the knowledge graph are not required. As the vocabulary size for both relation and type should be sufficiently small to reduce the total training parameters, a good filtering method is required to remove commonly-used and rare types linked to only one entity. For example, `Person` contains lesser information compared

to `Male_characters_in_film` as the former type can be found in most entities. Therefore, we proposed an entropy-based filtering to reduce the size of type "vocabulary". As the entropy value of types indicates the amount of the information that a linked entity can provide, we calculated the sum of entity entropy of a type based on the entity conditional probability on types using:

$$H_{t_j}(e, R) = -\frac{1}{N} \sum_{(e_i, t_j) \in R} p(e_i|t_j) log\ p(e_i|t_j), \tag{7}$$

where R is the set of valid entity-type pairs found in the knowledge graph, N is the total numbers of entity-type pairs, and $p(e_i|t_j)$ is the conditional probability of observing the known entity e_i given a type t_j. The conditional probability is equal to the reciprocal of the edge degree of a given type node. Hence, the entropy increases with an increase in the edge degree of a type node. It is worth noting that [5] uses a similar conditional entropy-based filtering to build a diverse conversation data. In contrast to the study of [5], we added a denominator N to the entropy for penalizing common type such as `Person`, `Thing`. To the best of our knowledge, this is the first study that applies an entropy-based filtering for trimming knowledge graph nodes.

4 Experimental Results

4.1 Experimental Settings

Datasets. We evaluated the performance of the proposed model on six benchmark datasets, including AIDA-CoNLL [13], MSNBC [6], AQUIANT [20], ACE2004 [26], WNED-ClueWEB [11] and WNED-Wiki datasets [11]. In addition, according to the study of [9], we utilized the top-30 entity candidates for all datasets evaluated using the prior $p(e|m)$ computed from Wikipedia and Yago datasets. The statistics for all six datasets can be found in the appendix.

Implementation Details. We choose to use the transformer architecture of Roberta $_{base}$[18] as the encoder model while using a significantly smaller Transformer for the decoder since the mentions size is small compared to their document length. We initialized the encoder parameters by Roberta $_{base}$ and entity embeddings parameters by the parameters pre-trained on link prediction task of Dbpedia knowledge graph.[3] Meanwhile, the decoder parameters are initialized following the previous work [29]. All models are trained on Wikipedia datasets for 250,000 iterations. Mixed precision training is used to speed up training while retaining a large batch size during each feed-forward pass. We implement our models with Pytorch 1.7 and run experiments on an RTX 3090 GPU. The source code and trained models will be publicly available.

Baselines. We compare CMEM-KG with the following state-of-the-art methods. 1) **DCA** [32]: a global entity disambiguation model which constructs a dynamic context of entities through an iterative process, containing two versions of supervised learning (**DCA-SL**) and reinforcement learning (**DCA-RL**) to explore all

[3] The statistics of the knowledge graph can be found in the appendix.

	Train	Micro-F1
DCA-SL	✓	89.34±0.59
DCA-RL	✓	88.72±0.32
roberta-GCN	✓	87.34 ±0.05
LUKE+IT	✓	87.06±0.23
LUKE+IT+KG	✓	87.64±0.15
CMEM-KG (no-kg)	✓	84.23± 0.19
CMEM-KG (avg)	✓	90.57±0.04
CMEM-KG (attn)	✓	**90.70** ±0.14
roberta-GCN		68.39
LUKE		62.27
LUKE+IT+KG		66.87
CMEM-KG (no-kg)		68.97
CMEM-KG (avg)		85.66
CMEM-KG (attn)		86.46

Table 1. Micro-F1 on the CoNLL validation dataset. Train marks the model is trained on the CoNLL training dataset.

possible linking results. 2) **Roberta+GCN**, which extends [15] by adding graph convolution network (GCN) [27] to generate features for the knowledge graph. 3) **LUKE+IT** [15], which modifies [31] to tackle global entity disambiguation through an iterative decoding algorithm maximizing the prediction confidence score, together with the pretrained Roberta $_{base}$ encoder. 4) **LUKE+IT+KG**, a variant of **LUKE+IT** that includes our proposed KG regularization.

To ensure a fair comparison, we re-implement all baselines under the same version of English Wikipedia dump (2020-04-20) to prevent domain gap advantage according to [14]. Moreover, a set of entity vocabularies consisting of 274,409 entities is used in all experiments. By sharing the same set of entity vocabularies, all baselines have the same top candidate sets for each mention. The embedding dimension is set to 300, and the number of training iterations is 250,000. Adam is adopted as the optimizer with a learning rate of 1e-4 during 10,000-step warm-up and, followed by a linear decay.

4.2 Quantitative Results

The performance of the CMEM-KG on CoNLL-test dataset was compared to those of other entity disambiguation models, and the results are shown in Table 1 in terms of the micro-F1 scores. CMEM-KG with link prediction regularization (*i.e.*, CMEM-KG (avg) and CMEM-KG (attn)) outperformed other disambiguation models. Furthermore, the performance of the CMEM-KG trained only on Wikipedia-based annotations without training on the in-domain training set of the CoNLL dataset was significantly higher than those of other models by at least 26.4% in terms of micro-F1. This could be attributed to the fact that the link prediction regularization had a positive influence on the architecture of the CMEM-KG; thus, preventing the overfitting of the model at the initial stage.

	CWEB	MSNBC	ACE2004	Wiki	AQUAINT	Avg
Prior $p(e\|m)$	**74.81**	75.06	76.15	73.72	76.40	75.23
DCA-RL	70.67	93.50	88.29	70.39	83.36	81.24
DCA-SL	72.75	93.04	87.81	75.21	84.81	82.74
roberta-GCN	69.81	91.10	86.87	71.85	81.15	80.16
LUKE+IT	72.57	90.59	89.90	76.21	84.81	82.82
LUKE+IT+KG	72.71	90.64	90.10	76.40	84.81	82.93
CMEM-KG (no-kg)	67.85	87.47	91.37	75.41	83.34	81.09
CMEM-KG (avg)	69.33	**93.58**	90.19	73.156	83.69	81.99
CMEM-KG (attn)	72.01	91.25	**92.69**	**76.44**	**88.30**	**84.14**

Table 2. Average Micro-F1 scores on WNED-ClueWEB (CWEB), MSNBC , ACE 2004, WNED-Wiki (Wiki), AQUAINT, each score is averaged from five different runs with different random seeds. The best results are shown in bold.

Table 2 shows the comparison of the performance of CMEM-KG and those of the other entity disambiguation models on the other five datasets. All the models listed in Table 2 were fine-tuned based on CoNLL datasets, and their hyper-parameters were tuned using the CoNLL validation set. The results revealed that the proposed CMEM-KG outperformed the baselines; particularly, on ACE2004 and AQUAINT datasets. This could be attributed to the fact that CMEM-KG with a link regularization prevented the overfitting of long-tailed mentions. However, the performance of CMEM-KG on CWEB was slightly poor which could be attributed to the fact that the automatic generation of the label resulted in noisy mentions. As pointed out by [32], CWEB also contains cases where none of the candidates are the actual answer which cause our model to predict the wrong answers. Moreover, adding link prediction regularization to LUKE+IT only slightly improves the overall average micro-F1, since it does not separate entity and word domain using different encoders.

4.3 Ablation Study

Impact of Aggregate Function. We evaluate the performance of two proposed aggregation functions (avg and attn) against the original TransE model by the link prediction on Dbpedia test dataset. Table 3 shows that adding aggregation function to model one-to-many relations improves link prediction with TransE+avg and TransE+attn performs better than TransE baseline in terms of Mean Reciprocal Rank, Hit@3, and Hit@10. The results indicate that our proposed attention mechanism performs better than the average function in all evaluation metrics, since they are able to learn distinct entity embedding with the help of fine-grained types.

Impact of Mention Frequency. To investigate the performance of our model on rare entities, following the setting of previous work [15] method, we first calculate the frequency for each entity found in Wikipedia dataset and split into 4 bins according to the frequency. Afterward, we calculate the Micro-F1 scores of

Dbpedia	MRR	Hit@3	Hit@10
TransE	16.5	13.9	29.2
TransE+avg	16.8	14.8	31.0
TransE+attn	19.3	17.4	40.7

Table 3. Comparison of link prediction results on Dbpedia test dataset.

CoNLL test-B split by bins. Table 4 shows the proposed CMEM-KG outperforms LUKE+IT, especially in predicting rare entities.

#	CMEM-KG		LUKE+IT	
annotations	attn	avg	confidence	natural
0	**68.37**	58.95	54.90	53.78
1-10	**93.42**	93.61	91.92	91.03
11-50	**89.66**	91.30	89.54	89.54
≥ 51	**91.26**	90.13	89.91	89.51

Table 4. Micro-F1 performance of the CMEM-KG on CoNLL test dataset split using Wikipedia entity frequency. All models were finetuned based on CoNLL dataset.

4.4 Evaluations on Iterative Entity Disambiguation

Iterations (N)	CoNLL-test	CWEB	MSNBC	ACE2004	Wiki	AQUIANT
3	17.36	-1.18	-6.01	10.44	-4.76	15.98
5	23.64	12.08	0.23	25.09	0.85	11.03
M	23.84	10.70	0.02	25.09	0.46	11.03

Table 5. The Micro-F1 percentage difference compared to parallel entity disambiguation (N=1) of CMEM-KG on all 6 datasets when decoding using N iterations. Results are averaged from 5 runs of different random seeds.

Following a similar approach by [10], we extend parallel entity disambiguation to N decoding iterations. For each iteration, the model only disambiguates the top-K confidence mentions, where $K = \frac{M}{N}$ and M is the total number of mentions. Each step is repeated until all mentions are resolved. Note that $N = 1$ equals to the parallel entity disambiguation used by our model while $N = M$ equals to the same confidence order proposed by [15].

Table 5 shows that splitting the parallel decoding into K iterations only provides slight improvements (less than 0.3%) as compared to our original parallel entity disambiguation ($N = 1$). Since most documents only have one mention to resolve, the iterative decoding leads to a minor improvement even when $N = M$. This suggested that our iterative decoding was more effective when there was

# mentions in context	N=1	N=3	N=5	N=7
1	86.10	86.10	86.10	86.10
$2 \leq M < 5$	86.54	**86.69**	86.69	86.69
$5 \leq M < 10$	88.46	88.46	**88.65**	88.65
$10 \leq M$	78.88	79.26	**79.63**	79.63

Table 6. Micro-F1 performance of the CMEM-KG model on CoNLL dataset based on the number of mentions for decoding iterations

more than one mention in the documents. To verify this, Table 6 shows the comparison of the performance of CMEM-KG on CoNLL dataset with different number of mentions in a document and different number of iterations. The results indicate that increasing the number of iterations improved the performance of the model on documents with multiple mentions, suggesting that five iterations are sufficient to handle documents with a large number of mentions.

5 Conclusion

In this study, we proposed a new global entity disambiguation model (CMEM-KG) and a new regularization method to train CMEM-KG. The proposed CMEM-KG only required one feed-forward pass to disambiguate mentions found in a given text. Moreover, the proposed regularization method prevented entity overfitting during training using a link prediction. Furthermore, we employed an entropy-based filtering to reduce the additional computation introduced by the regularization method to prune the nodes in the knowledge graph. Experimental results revealed that the proposed CMEM-KG is effective for different architectures and datasets. In the future, we plan to use a knowledge graph constructed by using relations between mentions from context instead of a predefined knowledge graph.

Acknowledgements

This work is supported in part by the Ministry of Science and Technology (MOST) of Taiwan under the grants MOST-109-2221-E-009-114-MY3 and MOST-110-2221-E-001-001. This work was also supported by the Higher Education Sprout Project of the National Yang Ming Chiao Tung University and Ministry of Education (MOE), Taiwan. We are grateful to the National Center for High-performance Computing for computer time and facilities.

References

1. Bordes, A., Usunier, N., Garcia-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPS (2013)
2. Broscheit, S.: Investigating entity knowledge in BERT with simple neural end-to-end entity linking. In: CoNLL (2019)

3. Chen, H.W., Shuai, H.H., Wang, S.D., Yang, D.N.: Quality-aware streaming network embedding with memory refreshing. In: PAKDD (2020)
4. Cheng, S.Y., Chen, Y.L., Yeh, M.Y., Lin, B.T.: Exploiting relevant hyperlinks in knowledge base for entity linking. In: PAKDD (2021)
5. Csáky, R., Purgai, P., Recski, G.: Improving neural conversational models with entropy-based data filtering. In: ACL (2019)
6. Cucerzan, S.: Large-scale named entity disambiguation based on wikipedia data. In: EMNLP-CoNLL (2007)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
8. Fatma, N., rVijay Choudhary, Sachdeva, N., Rajput, N.: Canonicalizing knowledge bases for recruitment domain. In: PAKDD (2020)
9. Ganea, O.E., Hofmann, T.: Deep joint entity disambiguation with local neural attention. In: EMNLP (2017)
10. Ghazvininejad, M., Levy, O., Liu, Y., Zettlemoyer, L.: Mask-predict: Parallel decoding of conditional masked language models. In: EMNLP-IJCNLP (2019)
11. Guo, Z., Barbosa, D.: Robust named entity disambiguation with random walks. Semantic Web (2018)
12. He, Z., Liu, S., Li, M., Zhou, M., Zhang, L., Wang, H.: Learning entity representation for entity disambiguation. In: ACL (2013)
13. Hoffart, J., Yosef, M.A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., Weikum, G.: Robust disambiguation of named entities in text. In: EMNLP (2011)
14. van Hulst, J.M., Hasibi, F., Dercksen, K., Balog, K., de Vries, A.P.: Rel: An entity linker standing on the shoulders of giants. In: SIGIR (2020)
15. Ikuya Yamada, K.W., Shindo, H., Matsumoto, Y.: Global entity disambiguation with pretrained contextualized embeddings of words and entities. arXiv (2019)
16. Le, P., Titov, I.: Improving entity linking by modeling latent relations between mentions. In: ACL (2018)
17. Li, B.Z., Min, S., Iyer, S., Mehdad, Y., Yih, W.t.: Efficient one-pass end-to-end entity linking for questions. In: EMNLP (2020)
18. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. ArXiv (2019)
19. Miao Fan, Qiang Zhou, E.C., Zheng, T.F.: Transition-based knowledge graph embedding with relational mapping properties. In: PACLIC (2014)
20. Milne, D., Witten, I.H.: Learning to link with wikipedia. In: CIKM (2008)
21. Mulang, I.O., Singh, K., Prabhu, C., Nadgeri, A., Hoffart, J., Lehmann, J.: Evaluating the impact of knowledge graph context on entity disambiguation models. In: CIKM (2020)
22. Ning, Z., Qiao, Z., Dong, H., Du, Y., Zhou, Y.: Lightcake: A lightweight framework for context-aware knowledge graph embedding. In: PAKDD (2021)
23. Pershina, M., He, Y., Grishman, R.: Personalized page rank for named entity disambiguation. In: NAACL-HLT (2015)
24. Pham, M.T.X., Cao, T.H., Huynh, H.M.: Candidate searching and key coreference resolution for wikification. In: IMCOM (2016)
25. Phan, M.C., Sun, A., Tay, Y., Han, J., Li, C.: Pair-linking for collective entity disambiguation: Two could be better than all. In: TKDE (2019)
26. Ratinov, L.A., Roth, D., Downey, D., Anderson, M.: Local and global algorithms for disambiguation to wikipedia. In: ACL (2011)

27. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks (2018)
28. Sevgili, Ö., Panchenko, A., Biemann, C.: Improving neural entity disambiguation with graph embeddings. In: ACL (2019)
29. Vaswani, A., Shazeer, N.M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. NIPS (2017)
30. Wainwright, M.J., Jordan, M.I.: Graphical models, exponential families, and variational inference. Found. Trends Mach. Learn. (2008)
31. Yamada, I., Asai, A., Shindo, H., Takeda, H., Matsumoto, Y.: LUKE: Deep contextualized entity representations with entity-aware self-attention. In: EMNLP (2020)
32. Yang, X., Gu, X., Lin, S., Tang, S., Zhuang, Y., Wu, F., Chen, Z., Hu, G., Ren, X.: Learning dynamic context augmentation for global entity linking. In: EMNLP-IJCNLP (2019)

A Supplementary Materials

A.1 Statistics of Datasets

Datasets	number mentions	number docs	mention per docs	Gold Recall
AIDA-train	18444	942	19.58	99.15
AIDA-test A	23232	216	107.55	99.10
AIDA-test B	27714	230	120.50	99.07
ACE2004	27954	35	798.69	99.06
AQUIANT	28651	50	573.02	99.06
CLUEWEB	39623	320	123.82	98.12
MSNBC	40272	20	2013.6	98.15
Wikipedia	46870	318	147.39	97.80

Table 7. Data statistics. Gold recall is the percentage of entities in top-30 candidates.

The statistics of six used datasets can be found in Table 7. Please note that Gold-Recall is the percentage of candidate set containing the ground truth entity. The Gold-Recall can be viewed as the upper bound limit for Micro-F1 score.

Dataset	Dbpedia				
Entities	274,476	**Relations**	2,389	**Types**	223,817
Training Triplets	1,146,240	**Validation Triplets**	181,597	**Testing Triplets**	257,392

Table 8. Statistics of the KG constructed from Dbepdia.

A.2 Statistics of the knowledge graphs

We initialize the encoder parameters by Roberta $_{base}$ and entity embeddings parameters by the parameters pre-trained on link prediction task of Dbpedia knowledge graph. The statistic of the Dbpedia knowledge graph can be found in Table 8.

A.3 An Illustrative Example of Entity Entropy

Type	Entropy	Linked entity
Wikicat1956FilmFestivals	0	1
Thing	0.0001	130771
Person	0.0003	31368
American_culture	0.1107	31
WikicatModernistComposers	0.244	9
Films_with_screenplays_by_John_Musker	0.366	3

Table 9. Statistic of a selection of types found in knowledge graph

Table 9 shows the entropy of commonly-found types such as **Thing**, **Person**, which are penalized to a small value. The entropy of rare types only linking to one entity is 0, such as **Wikicat1956FilmFestivals**. By removing types with entropy smaller than 0.1, we 57.25% of type nodes can be trimmed from the knowledge graph. In this case, the threshold removes type nodes with the edge degree lower than 2 or higher than 13.