HIGH TECHNOLOGY LETTERS | Vol. 30 No. 1 | Mar. 2024 | pp. 13-22

doi:10.3772/j.issn.1006-6748.2024.01.002

SHEL: a semantically enhanced hardware-friendly entity linking method①

QI Donglin (亓东林), CHEN Shudong②, DU Rong, TONG Da, YU Yong
(Institute of Microelectronics of Chinese Academy of Sciences, Beijing 100029, P. R. China)
(University of Chinese Academy of Sciences, Beijing 100190, P. R. China)

Abstract

With the help of pre-trained language models, the accuracy of the entity linking task has made great strides in recent years. However, most models with excellent performance require fine-tuning on a large amount of training data using large pre-trained language models, which is a hardware threshold to accomplish this task. Some researchers have achieved competitive results with less training data through ingenious methods, such as utilizing information provided by the named entity recognition model. This paper presents a novel semantic-enhancement-based entity linking approach, named semantically enhanced hardware-friendly entity linking (SHEL), which is designed to be hardware friendly and efficient while maintaining good performance. Specifically, SHEL's semantic enhancement approach consists of three aspects: (1) semantic compression of entity descriptions using a text summarization model; (2) maximizing the capture of mention contexts using asymmetric heuristics; (3) calculating a fixed size mention representation through pooling operations. These series of semantic enhancement methods effectively improve the model's ability to capture semantic information while taking into account the hardware constraints, and significantly improve the model's convergence speed by more than 50% compared with the strong baseline model proposed in this paper. In terms of performance, SHEL is comparable to the previous method, with superior performance on six well-established datasets, even though SHEL is trained using a smaller pre-trained language model as the encoder.

Key words: entity linking (EL), pre-trained models, knowledge graph, text summarization, semantic enhancement

0 Introduction

Entity linking (EL) is the process of linking entity mentions in a text to their corresponding entities in a knowledge base, such as Wikipedia or Freebase. This process aims to resolve the ambiguity of entity mentions, which may refer to different entities in different contexts. Entity linking related algorithms have developed rapidly in recent years and play an important role in knowledge engineering and data mining, underlying a variety of downstream applications such as knowledge base population, content analysis, relation extraction, and question answering[1]. In recent years, several high-performance entity linking models have emerged, mainly due to two reasons[2]: (1) encoding with pre-trained language models enhances the semantic capture of entity linking models; (2) training on larger amounts of data improves the generalization ability of entity linking models. Many entity linking approaches fine-tune the model for better performance by using large scale pre-trained models such as the bidirectional encoder representations from transformers-large (BERT-large)[3] as the text encoder. Even with the less computationally intensive bi-encoder scheme, which separately encodes mention contexts and the entity descriptions, these methods often require more than 16 GB GPU memory size, making them difficult to be trained on most consumer graphics cards. In addition to using large pre-trained models, some entity linking methods are trained on large amounts of data to improve performance, such as the generative entity retrieval (GENRE)[4] model. This poses a challenge for researchers with limited hardware resources, who have to resort to using smaller pre-trained models and less data for entity linking, resulting in a significant loss of performance. To address

① Supported by the Beijing Municipal Science and Technology Program (Z231100001323004).
② To whom correspondence should be addressed. E-mail: chenshudong@ime.ac.cn.
Received on June 20, 2023

this problem, some researchers have proposed the method of utilizing named entity recognition (NER) for entity linking, which achieves competitive results with less training data. In addition, it has been observed that there is a limit to the length of the input sequence when using pre-trained language models as encoders, especially when encoding entity descriptions, some of which can consist of thousands of tokens, far exceeding the maximum input length of most pre-trained models, and it is common to truncate these extra-long texts according to a predetermined maximum sequence length. To mitigate the semantic loss of truncated text, MuVER[5], an entity linking model that models each sentence individually, has been proposed to improve the performance of the entity linking task in the zero-shot scenario by modelling text at a smaller granularity.

Inspired by the above, semantically enhanced hardware-friendly entity linking (SHEL) is presented, a hardware-friendly and efficient entity linking method that uses semantic enhancement. SHEL uses a text summarization model for semantic compression of the entity descriptions, employs asymmetric heuristics to maximize the capture of mention contexts, and uses the fixed size mention representation in combination to exploit as much semantic information as possible, with the expectation that the model will capture useful semantic features more easily. By using small scale pre-trained models as encoders, experiments have shown that SHEL can be efficiently trained on common hardware. Furthermore, SHEL outperforms the corresponding baseline model without semantic enhancement in most cases, with a training speedup of more than 50%. Even when compared with entity linking methods using larger pre-trained models as encoders, SHEL is still competitive in terms of performance.

Above all, the main contributions are as follows.

(1) A semantic-enhancement-based hardware-friendly and efficient entity linking method, SHEL, is proposed. This method mainly consists of three parts, i. e. the semantic compression of entity descriptions, the asymmetric heuristic to maximize the capture of mention contexts, and the fixed size mention representation.

(2) The method can achieve entity linking performance comparable to that based on larger scale pre-trained language models as the encoder. At the same time, the training speed can be doubled using this method.

(3) A wider range of applications for text summarization models is explored. In the era of data explosion, text summarization models can not only help humans to extract important semantic information, but also act as a text compression method to help models extract semantics and reduce hardware requirements. With the emergence of large models, using the powerful semantic compression capability of these models to help small models select training data is a good solution to balance the computational deployment overhead and model performance.

1 Related work

1.1 Entity linking

In recent years, entity linking methods based on pre-trained language models have gradually become mainstream and have achieved state-of-the-art results for this task, due to the powerful semantic extraction and characterisation of text by pre-trained language models. When using pre-trained models to encode text data for the entity linking task, there are two types of encoding methods: bi-encoder and cross-encoder. The bi-encoder trains separate encoders for the mention context and the entity description, while the cross-encoder combines related contexts of the mention and the entity description into one whole for encoding. The cross-encoder introduces many fine-grained interactions between the mention contexts and the entity description, which is more accurate[6], while the bi-encoder has the advantage of being faster and consuming less hardware resources. Assuming that N mentions are linked to a knowledge base containing M entities, the encoding times required for the bi-encoder is $N + M$, while the encoding times for the cross-encoder is $N \times M$. Due to the large number of entities in the knowledge base, the value of M can often reach 6 million, such as Wikipedia, or even more, making the computational overhead of encoding times unacceptable. Besides, splicing mention contexts and entity descriptions results in longer text to be encoded, and due to the input length limit of the pre-trained language model, more text content is truncated beyond this length limit compared with the bi-encoder approach. To improve the performance of entity linking models, researchers have experimented with various pre-trained models such as BERT[3], SpanBERT[7] and BioBERT[8]. Larger scale language models based on more corpus training tend to be more advantageous when applied to downstream tasks, and the same is true for the entity linking task. However, it is undeniable that as the number of parameters of the pre-trained model increases, so does the hardware overhead for just fine-tuning the pre-trained model, requiring even top-of-the-range consumer graphics cards.

With the help of the powerful semantic representa-

tion capabilities of pre-trained models, some researchers have proposed GENRE[4], a generative model trained on the KILT[9] dataset with massive data, about 9 million training instances, to perform the entity linking task, marking the emergence of generative entity linking models. Unlike retrieval-based entity linking, which selects the most likely entity from the knowledge base as the correct one to link to, generative entity linking restricts the generated content by generating a unique name for the entity to be linked directly based on the entities in the established knowledge base. Generative models tend to have better generalisation performance while achieving high link accuracy, but they also have an exponentially higher hardware overhead compared with retrieval-based models, and larger models also result in longer training times.

In addition to the above two types of methods for improving model performance, some researchers have tried to improve the performance of entity linking in other ingenious ways. One type of approach is to introduce external knowledge sources, such as introducing structural information from knowledge graphs[10] or introducing an named entity recoanition (NER) model to aid discrimination[1]. Another type of approach is to do joint training of entity linking with other tasks, the most common of which is the coreference resolution[11-12]. There is also a type of approach that attempts to improve the linking capability by improving the way the model encodes the data, such as MuVER, which does not rely on text encoding for the entity descriptions and employs a more fine-grained sentence encoding. However, whether introducing external knowledge sources, co-training with other tasks or encoding finer-grained sentences, all these approaches are additive to the training dataset, increasing the cost of training the model. The approach proposed in this paper goes in the opposite direction, exploring possible high performance entity linking solutions without adding additional information or even compressing the training data.

1.2 Text summarization

Text summarization is the process of using natural language process (NLP) techniques to transform an original text document into a shorter piece of text that highlights the most important information in the document according to a given criterion[13], and has made significant progress over the last decade. Text summarization can be divided into two main approaches: extractive and abstractive. Extractive summarization selects the most relevant sentences or phrases from the original text and links them together to form a summary. This type of approach uses various techniques such as graph-based algorithms, clustering and ranking algorithms. Abstractive summarization generates summaries by rephrasing and reorganizing the original text. This type of approach requires a higher level of natural language understanding and generation and is therefore more difficult than extractive summarization. Some of the important models in this category are transformer-based models such as the bidirectional and auto-regressive transformers (BART)[14], T5[15], and the generative pre-trained transformer-family (GPT-family) models[16-18], which have shown significant results.

Text summarization models have a wide range of applications, such as news summarization, document summarization, and search engine result summarization. By providing accurate and concise summaries of large amounts of text in various domains, text summarization models can save labor time and increase efficiency.

In this paper, text summarization models can be used not only to serve humans but also to serve machines better, i. e. using text summarization models to compress text data and reduce model training time and hardware overhead. In addition, for tasks with large amounts of text data and relatively low information density, the use of text summarisation also serves to remove noise and stop words, which can further improve the model performance.

2 Method

2.1 Problem setup

Given a knowledge base (KB) containing a set of entities with titles and descriptions, the goal of entity linking is to identify multiple mentions within a given document, which are typically marked in order, and link them to their corresponding entities in the KB. This is achieved through a function $M: M \rightarrow E$, where M represents mentions and E represents the set of entities.

2.2 Baseline system

A bi-encoder architecture is used, similar to the work that uses NER for entity linking (NER4EL)[1], but modified to switch the pre-trained model from BERT-large to the hardware-friendly model BERT-base. Mention contexts are represented as

$$[\mathrm{CLS}]\text{left contexts}[\mathrm{E}]\text{mention}[/\mathrm{E}]\text{right contexts}[\mathrm{SEP}] \quad (1)$$

where the left contexts and right contexts represent the left and right contexts of at most 64 tokens of the mention. The representation of the mention is taken from the embedding of the first token in the mention after encoding, and the corresponding entity is represented as

$$\text{ner\_tag}[\mathrm{NER\_SEP}]\text{description} \quad (2)$$

where description denotes the entity description, following NER4EL, and ner_tag denotes the NER category corresponding to the entity. The representation of the entity is taken from the embedding corresponding to the first token of the encoded description. As in previous work[19], each entity is modelled by taking the first 128 tokens of the corresponding Wikipedia article as input .

2.3 Semantic compression of entity descriptions

In the baseline system presented in subsection 2.2, the text truncation for the overly long entity descriptions (the corresponding Wikipedia articles), which truncates more than a third of the entity descriptive texts as shown in Fig. 1, results in a loss of semantics in the encoding of the entity.

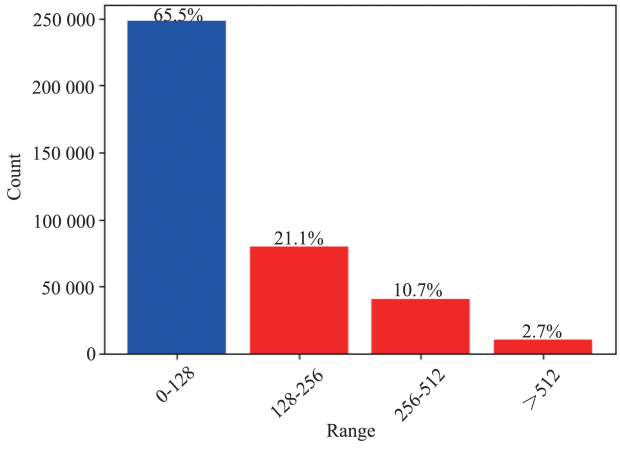

Fig. 1 Token length of entity descriptions related to the AIDA-YAGO-CoNLL dataset

As described in subsection 2.2, entity descriptions with more than 128 tokens are truncated due to the maximum encoding length limit. Therefore, the first column in the diagram represents the full entity descriptions that can be entered into the encoder, while the second, third and fourth columns represent the truncated entity descriptions . Over a third of the text is truncated. Some researchers have noted the problem that previous approaches are good at managing entities with short descriptions[6], but retrieving entities with long descriptions seems cumbersome because long descriptions contain too much information to be encoded into a single fixed-size vector. MuvER[5] has been proposed to encode each sentence of an entity description individually, significantly reducing the length of text to be encoded. However, this makes the encoding times grow exponentially, increasing the model complexity and computational overhead, which goes against the original intention of this paper. To this end, a text compression method based on the text summarization model is proposed to preserve as much semantic information as possible for entity descriptions longer than the fixed length. To achieve the semantic compression of the text, the classic model BART-large-CNN[14] is chosen, which compresses the long text of the entity descriptions to approximately 128 tokens and then inputs them into the BERT-base model for encoding. This not only effectively controls the number of tokens to be encoded in the model and ensures encoding efficiency, but also requires very little truncation of the processed text, reducing the semantic loss compared with truncating the original text directly. For a total of 110 000 items of entity descriptions related to the AIDA-YAGO-CoNLL dataset to be compressed, the token length variation statistics are shown in Table 1, where all data statistics are for entity descriptions to be compressed with a fixed length, not for all entity descriptions, with the entity representation adjusted to

$$\text{title}\,[\,\text{TITLE\_SEP}\,]\,\text{ner\_tag}\,[\,\text{NER\_SEP}\,]\,\text{compressed\_description} \quad (3)$$

where title refers to the unique name of the entity, compressed_description refers to the compressed description of the entity.

Table 1 Token length statistics of entity descriptions related to the AIDA-YAGO-CoNLL dataset

	Exceeding 128 tokens	Minimum token length	Maximum token length	Average token length
Before compression	111 053	129	6 432	267.5
After compression	18 843	79	164	123.6

2.4 Asymmetric heuristic maximizes capture of mention contexts

In the baseline system presented in subsection 2.2, the mention is encoded with a symmetric length for the left and right contexts around mentions. When there is a large difference in the number of tokens in the left and right contexts around mentions, a large amount of padding is used to keep the number of input tokens constant. Some researchers have noticed this and adopted a context asymmetric acquisition approach[5], where the number of left and right contexts around mentions is adjusted by rules to capture as much as possible within a given number of tokens. However, this approach ignores the common knowledge that the core idea of a paragraph is often found in the first sentence. When the mention is located later in the text of the paragraph, using this method will result in insufficient capture of

the overall semantics of the text. An improved method is proposed, and the pseudocode for fetching mention contexts is shown as Algorithm 1.

Algorithm 1 Asymmetric heuristic maximizes capture of mention contexts

Input: Left contexts around the mention after being tokenized named as *left_conterts_tokenized*, the mention after being tokenized named as *mention_tokenized*, and right contexts around the mention after being tokenized named as *right_conterts_tokenized*, maximum input length for pre-trained models named as *sequence_max_length*

```
left_quota←(sequence_max_length − LEN(mention_tokenized))//2 − 1
right_quota←sequence_max_length − LEN(mention_tokenized) − leftquota − 2
left_origin←LEN(left_conterts_tokenized)
right_origin←LEN(right_conterts_tokenized)
if left_origin < = left_quota then
    if right_origin > right_quota then
        right_quota←right_quota + left_quota − left_origin
    end if
else
    if right_origin < = right_quota then
        lift_quota←left_quota + right_quota − right_origin
    end if
end if
mention_tokenized_total←tokenized_left_context[: left_quota] + mention_tokenized + tokenized_right_context[right_quota]
```

Output: Truncated mention contexts after being tokenized named as *mention_tokenized_total*

2.5 Fixed size mention representation

A detailed discussion of mention representation has been given in Ref. [20]. In the baseline system presented in subsection 2.2, the embedding corresponding to the first token of the mention is used as the mention representation, which is also known as the 'head embedding'. The end-to-end entity linking (EL) model[20] incorporated the embeddings of the first, last and the 'soft head' words of the mention to form a more complex representation of the mention based on an attention mechanism. Since a mention usually consists of two tokens, the performance gain of the 'soft head' is very limited. In order to balance the model performance and the computational overhead, a compromise scheme is proposed by pooling the 'head embedding' and the 'end special token embedding', as shown in Fig. 2. The 'head embedding' contains the most important semantic information of the mention, while the 'end special token embedding', which is located after the entire mention, contains the most complete semantic information. Experimental results demonstrate that using a combination of the two can lead to model performance gains.

2.6 Combinations of semantically enhanced contributions

All contributions can be combined to achieve further improvements. Of course, you can choose any two of them to improve the performance of the model as needed, and they do not conflict with each other. The model after introducing all the contributions on top of the baseline is called SHEL (semantically enhanced hardware-friendly entity linking). As shown in Fig. 3, the execution process of SHEL is as follows.

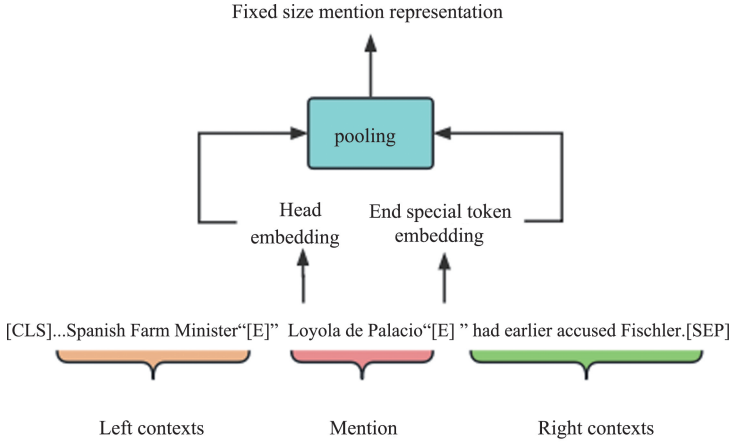

Fig. 2 Fixed size mention representation

(1) Preprocessing of candidate entity descriptions in KB, i.e. for entity descriptions with more than 128 tokens, semantic compression is performed to make them as much as possible within the range of 128 tokens.

(2) The left and right contexts around mentions in the document are pre-processed, and as much context

information as possible is obtained for a given token length by asymmetric heuristic.

(3) The results of the above two pre-processing steps are encoded using the BERT-based encoder, except that the entity description encoding is combined with the NER-related information, and the mention context encoding is pooled to obtain a fixed-size mention representation.

(4) For the encoding results of the two encoders in Step (3), the cosine similarity is calculated and the final entity linking results are obtained by combining the constraints due to the NER information.

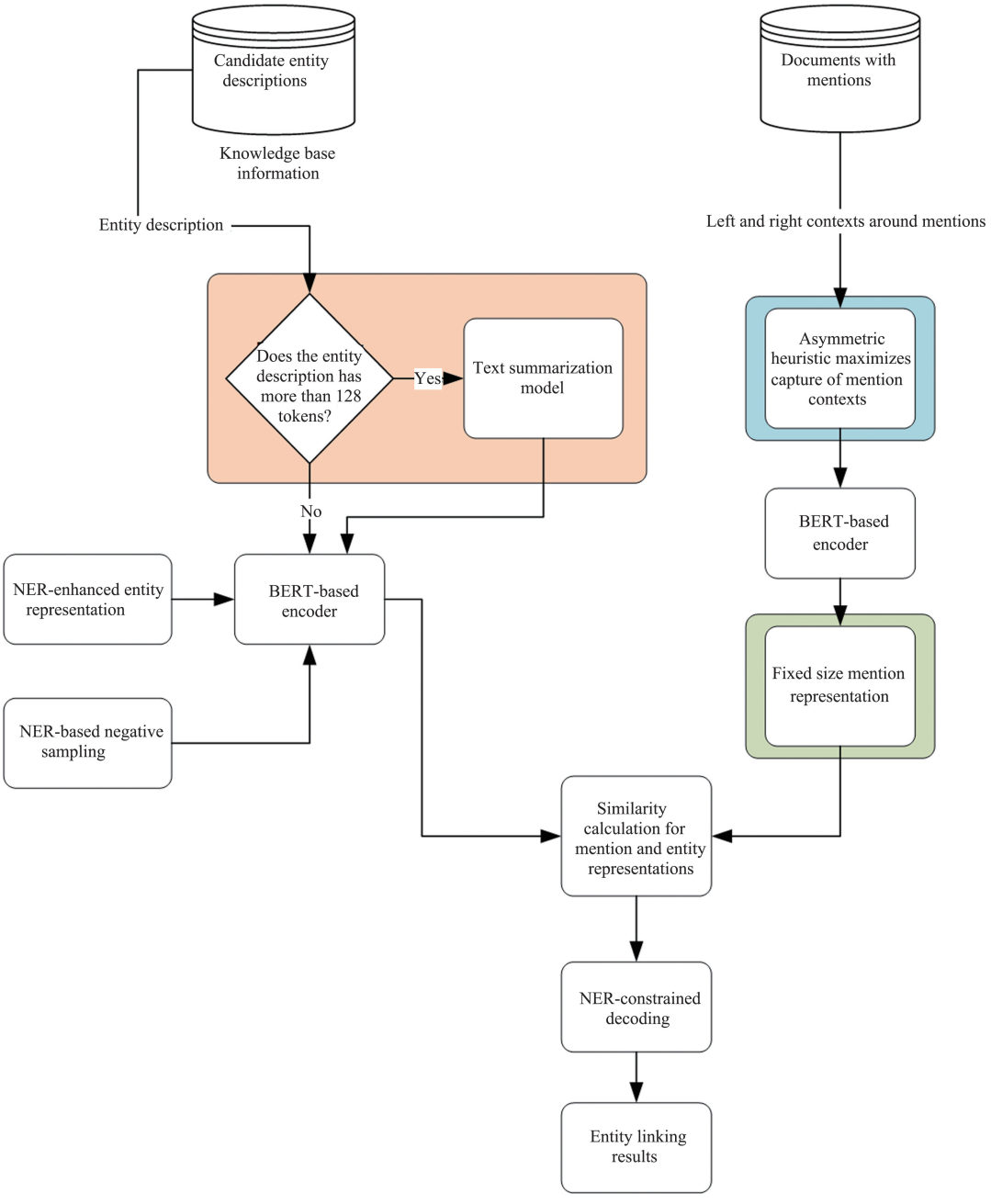

Fig. 3 SHEL model architecture

3 Experimentation and analysis

3.1 Datasets

The evaluation of the entity linking task is performed on the proposed models, following the experimental settings of previous work[4,21-22]. The proposed models were trained only on the AIDA-YAGO-CoNLL[23] training split, which contains 18 000 labelled instances. The proposed models are validated and tested on the corresponding parts of the datasets. For out-of-domain performance, the proposed model is measured

on other datasets including MSNBC[24], AQUAINT[25], ACE2004[26], WNED-CWEB[27], and WNED-WIKI[28]. Detailed statistical information on these datasets can be found in Table 2. For the AIDA-YAGO-CoNLL dataset, the statistics are divided into three parts: train, valid and test. All other datasets are used as test datasets only. Linkable mentions means that candidate entities of the mention can be found in the KB according to the alias table.

Table 2 Statistical data for six well-established datasets

Datasets	Documents	Mentions	Linkable mentions
AIDA-YAGO-CoNLL_train	946	18 395	17 821
AIDA-YAGO-CoNLL_valid	216	4 784	4 676
AIDA-YAGO-CoNLL_test	231	4 464	4 425
MSNBC	20	656	651
AQUAINT	50	743	720
ACE2004	57	259	256
WNED-WIKI	320	6 821	6 768
WNED-CWEB	320	11 154	11 077

3.2 Experimental setup

The mentions where no candidate entities could be found in the knowledge base are skipped directly. As shown in Table 2, in terms of the number of mentions and the number of linkable mentions, the skipped mentions represent a very small fraction of the whole dataset and do not have a significant impact on the final experimental results.

When using the text summarisation model to compress entity descriptions, as the maximum encoding length of the pre-trained model is set to 128, the output of the text summarisation model is controlled by setting parameters to stay within this range as much as possible, specifically the maximum output length is set to 130, the minimum output length is set to 120, and the penalty factor is set to 2.

The baseline of EL model and SHEL model and their variants are implemented in PyTorch, using the Transformation library[29] to load and fine-tune the BERT-base as the encoder. The proposed model is trained on configurations using Adam[30] with a learning rate of 10^{-4} and a weight decay of the same value of 10^{-4} for 100 epochs, implementing an early stopping strategy. The results of the best model checkpoints are reported, based on their $F1$ score obtained from the validation split of the AIDA-YAGO-CoNLL dataset. In terms of hardware, all models using BERT-base as the encoder were trained on the NVIDIA Titan Xp. The NER4EL model using BERT-large as the encoder required more GPU memory, so training was performed on the NVIDIA GeForce RTX 3090. In fact, the proposed models use no more than 8 GB GPU memory size and can be trained on a very basic deep learning platform.

3.3 Experimental results

The InKB micro-F1 score[31] used to evaluate the performance of the proposed model is reported. As shown in Table 3, the EL Baseline is the baseline system proposed in subsection 2.2. The Semantic Compression is the introduction of semantic compression of entity descriptions into the baseline system, as described in subsection 2.3. The Asymmetric Mention Contexts Extraction is the introduction of the asymmetric heuristic that maximises the capture of mention contexts into the baseline system as described in subsection 2.4. The Fixed Size Mention Representation is the introduction of the fixed size mention representation method into the baseline system as described in subsection 2.5. The SHEL model is the final model that incorporates all the contributions. In addition to this, the final results are tested and reported based on the open source code of NER4EL using default parameters to make a comparison with the model proposed in this paper and several variants, in particular, NER4EL uses BERT-large as the encoder, while the model proposed in this paper and all variants use the smaller pre-trained model BERT-base.

Experimental results are shown in Table 3. The results focus on models that use BERT-base as the encoder, with the best value in bold and the second best underlined. It can be seen that the three semantic enhancement methods proposed in this paper can improve model performance on both in-domain and out-of-domain datasets, and their performance improvements have different emphases. The model variants Semantic Compression, Asymmetric Mention Contexts Extraction and Fixed Size Mention Representation all outperform

the EL baseline, which is a very strong baseline for performance. When using BERT-base as an encoder, the Semantic Compression model tested on the ACE2004 dataset achieved the best results, the Asymmetric Mention Contexts Extraction model achieved the best results on the WNED-WIKI dataset, and the Fixed Size Mention Representation model achieved the best results on the MSNBC dataset. Interestingly, the final model SHEL, which combines all three semantic enhancement methods, achieved the best results of all models using BERT-base as the encoder on the other three datasets AIDA-YAGO-CoNLL, AQUAINT, WNED-CWEB, and a very competitive performance on the datasets ACE2004, WNED-WIKI, and degraded only on the dataset MSNBC. Even when compared with the model using BERT-large as the encoder, the performance of the SHEL model is competitive.

Table 3 InKB micro *F*1 scores on in-domain and out-of-domain test datasets

Encoder	Model	AIDA	MSNBC	AQUAINT	ACE2004	WNED-CWEB	WNED-WIKI
BERT-large	NER4EL	0.921	0.876	0.690	0.898	0.660	0.620
BERT-base	EL Baseline	0.895	0.848	0.656	0.848	0.636	0.592
	Semantic Compression	0.907	0.853	**0.672**	**0.891**	0.645	0.600
	Asymmetric Mention Contexts Extraction	0.908	0.857	**0.672**	0.871	0.655	**0.618**
	Fixed Size Mention Representation	0.902	**0.869**	0.661	0.879	0.647	0.604
	SHEL	**0.913**	0.839	0.672	0.875	**0.658**	0.617

As shown in Table 4, the results focus on models that use BERT-base as the encoder, with the best value in bold and the second best underlined, all proposed models and their variants have only a third of the number of training parameters of models using BERT-large as the encoder. With the same training data, all proposed models and their variants converge faster, with the model using BERT-large as the encoder requiring almost nine training epochs, while the proposed SHEL model converges in only three epochs. Due to the different hardware deployed for model training, the training time for the NER4EL model cannot be fairly compared to the model using BERT-base as the encoder. However, the EL Baseline differs from the NER4EL only in the encoder, so the training efficiency of the NER4EL model can be roughly estimated from the training time of the baseline. When making a comparison only within models that use BERT-base as the encoder, it can be seen that the three semantic enhancement methods (Semantic Compression, Asymmetric Mention Contexts Extraction and Fixed Size Mention Representation) and the final model SHEL converge faster than the EL Baseline. In particular, the Semantic Compression variant and the SHEL model converge in about half the training time and epochs of the baseline model. This is due to the high quality compression of the training data by the text summarization model, which improves the effective use of the data by the entity linking model. It is encouraging to note that the final model SHEL achieves very similar performance to the model NER4EL using BERT-large as the encoder for about 1/3 of the training data size and 1/3 of the training epochs. These experimental results provide ample evidence that the proposed model is both hardware-friendly and efficient while maintaining excellent performance.

Table 4 Statistics on the ease of hardware deployment and training efficiency of different models

Encoder	Model	Parameters/10^6	Training instances/10^3	AIDA accuracy	Training time/h	Number of epochs
	GENRE	406	9 000	0.933		
BERT-large	NER4EL	335	18	0.921		8.95
BERT-base	EL Baseline	109	18	0.895	20.87	6.76
	Semantic Compression	109	18	0.907	11.49	3.45
	Asymmetric Mention Contexts Extraction	109	18	0.908	15.36	5.00
	Fixed Size Mention Representation	109	18	0.902	14.57	4.76
	SHEL	109	18	**0.913**	**9.97**	**3.00**

4 Conclusions

This paper proposes an efficient and hardware-friendly entity linking method based on semantic enhancement, called SHEL, which consists of three components: (1) semantic compression of entity descriptions; (2) asymmetric heuristic to maximize capture of mention contexts; (3) fixed size mention representation. The SHEL method exploits semantic information to improve the accuracy of entity linking while minimising the computational requirements. Experimental results show that the proposed method offers superior performance and outstanding advantages in terms of hardware friendliness and efficiency compared to the strong baseline model.

With the development of text generation techniques, the use of a stronger text summarization model instead of the classical model (BART-large-CNN) in this paper should yield better results. In the meantime, it is worth exploring how to effectively compress the mention contexts, in addition to compressing the entity descriptions. Further, it is also worth exploring how text summarization models can be used to support other natural language processing related tasks with large amounts of text data.

References

[1] SHEN W, LI Y, LIU Y, et al. Entity linking meets deep learning: techniques and solutions[J]. IEEE Transactions on Knowledge and Data Engineering, 2021, 35(3): 2556-2578.

[2] TEDESCHI S, CONIA S, CECCONI F, et al. Named entity recognition for entity linking: what works and what's next [C]// Findings of the Association for Computational Linguistics. Punta Cana, Dominican: EMNLP, 2021: 2584-2596.

[3] KENTON J D M W C, TOUTANOVA L K. BERT: pre-training of deep bidirectional transformers for language understanding[C] // Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, USA: NAACL-HLT, 2019: 4171-4186.

[4] DE CAO N, IZACARD G, RIEDEL S, et al. Autoregressive entity retrieval [C] // International Conference on Learning Representations. Online: ICLR, 2021: 1-18.

[5] MA X, JIANG Y, BACH N, et al. MuVER: improving first-stage entity retrieval with multi-view entity representations[C]// Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Online: EMNLP, 2021: 2617-2624.

[6] WU L, PETRONI F, JOSIFOSKI M, et al. Scalable zero-shot entity linking with dense entity retrieval[C] // Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing. Punta Cana, Dominican: EMNLP, 2020: 6397-6407.

[7] XU L, CHOI J D. Revealing the myth of higher-order inference in coreference resolution[C] // Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing. Punta Cana, Dominican: EMNLP, 2020: 8527-8533.

[8] ANGELL R, MONATH N, MOHAN S, et al. Clustering-based inference for zero-shot biomedical entity linking[C] // Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Mexico City, Mexico: NAACL-HLT, 2021: 1-11.

[9] PETRONI F, PIKTUS A, FAN A, et al. KILT: a benchmark for knowledge intensive language tasks[C] // Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Mexico City, Mexico: NAACL-HLT, 2021: 2523-2544.

[10] SEVGILI Ö, PANCHENKO A, BIEMANN C. Improving neural entity disambiguation with graph embeddings[C] // Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: ACL, 2019: 315-322.

[11] ZAPOROJETS K, DELEU J, JIANG Y, et al. Towards consistent document-level entity linking: joint models for entity linking and coreference resolution[C] // Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics. Dublin, Ireland: ACL, 2022: 1-7.

[12] AGARWAL D, ANGELL R, MONATH N, et al. Entity linking via explicit mention-mention coreference modeling [C]// Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Seattle, USA: NAACL-HLT, 2022: 4644-4658.

[13] CAJUEIRO D O, NERY A G, TAVARES I, et al. A comprehensive review of automatic text summarization techniques: method, data, evaluation and coding[EB/OL]. (2023-01-04)[2023-06-20]. https://arxiv.org/pdf/2301.03403.pdf.

[14] LEWIS M, LIU Y, GOYAL N, et al. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension[C] // Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: ACL, 2020: 7871-7880.

[15] RAFFEL C, SHAZEER N, ROBERTS A, et al. Exploring the limits of transfer learning with a unified text-to-text transformer[J]. The Journal of Machine Learning Research, 2020, 21(1): 5485-5551.

[16] RADFORD A, NARASIMHAN K, SALIMANS T, et al. Improving language understanding by generative pre-training [EB/OL]. (2018-06-12)[2023-06-20]. https://gwern.net/doc/www/s3-us-west-2.amazonaws.com/d73fdc5ffa8627bce44dcda2fc012da638ffb158.pdf.

[17] RADFORD A, WU J, CHILD R, et al. Language models are unsupervised multitask learners[EB/OL]. (2019-02-14)[2023-06-20]. https://openai.com/blog/better-lang

uage-models.
[18] BROWN T, MANN B, RYDER N, et al. Language models are few-shot learners[J]. Advances in Neural Information Processing Systems, 2020, 33: 1877-1901.
[19] BOTHA J A, SHAN Z, GILLICK D. Entity linking in 100 languages[C]// Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing. Punta Cana, Dominican: EMNLP, 2020: 7833-7845.
[20] KOLITSAS N, GANEA O E, HOFMANN T. End-to-end neural entity linking[C]// Proceedings of the 22nd Conference on Computational Natural Language Learning. Brussels, Belgium: CoNLL, 2018: 519-529.
[21] GANEA O E, HOFMANN T. Deep joint entity disambiguation with local neural attention [C]// Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics. Copenhagen, Denmark: EMNLP, 2017: 2619-2629.
[22] LE P, TITOV I. Improving entity linking by modeling latent relations between mentions[C]// The 56th Annual Meeting of the Association for Computational Linguistics: Association for Computational Linguistics. Melbourne, Australia: ACL, 2018: 1-10.
[23] HOFFART J, YOSEF M A, BORDINO I, et al. Robust disambiguation of named entities in text[C]// Proceedings of the Conference on Empirical Methods in Natural Language Processing. Edinburgh, UK: EMNLP, 2011: 782-792.
[24] CUCERZAN S. Large-scale named entity disambiguation based on Wikipedia data[C]// Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Prague, Czech: EMNLP-CoNLL, 2007: 708-716.
[25] MILNE D, WITTEN I H. Learning to link with Wikipedia [C]// Proceedings of the 17th ACM Conference on Information and Knowledge Management. Napa Valley, USA: Association for Computing Machinery, 2008: 509-518.
[26] RATINOV L, ROTH D, DOWNEY D, et al. Local and global algorithms for disambiguation to Wikipedia[C]// Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Association for Computational Linguistics. Portland, USA: ACL, 2011: 1375-1384.
[27] EVGENIY G, MICHAEL R, AMARNAG S. Facc1: freebase annotation of clueweb corpora[EB/OL]. [2023-06-20]. https: // arxiv. org/ftp/arxiv/papers/1712/1712.08355. pdf.
[28] GUO Z, BARBOSA D. Robust named entity disambiguation with random walks[J]. Semantic Web, 2018, 9(4): 459-479.
[29] WOLF T, DEBUT L, SANH V, et al. Transformers: state-of-the-art natural language processing[EB/OL]. (2019-10-14)[2023-06-20]. https: // arxiv. org/pdf/1910.03771v2. pdf.
[30] DIEDERIK P. KINGMA J B A. Adam: a method for stochastic optimization[C]// The 3rd International Conference on Learning Representations. San Diego, USA: ICLR, 2015: 1-15
[31] RÖDER M, USBECK R, NGONGA N A C. Gerbil—benchmarking named entity recognition and linking consistently[J]. Semantic Web, 2018, 9(5): 605-625.

QI Donglin, born in 1996. He is a Ph. D candidate at the Institute of Microelectronics of the Chinese Academy of Sciences and the University of Chinese Academy of Sciences. He received his B. S. degree from University of Science and Technology Beijing in 2019. His research interests focus on natural language processing and knowledge graph.