Pre-training of Deep Contextualized Embeddings of Words and Entities for Named Entity Disambiguation

Ikuya Yamada[1,3]
ikuya@ousia.jp

Hiroyuki Shindo[2,3]
shindo@is.naist.jp

[1]Studio Ousia, Tokyo, Japan
[2]Nara Institute of Science and Technology, Nara, Japan
[3]RIKEN AIP, Tokyo, Japan

Abstract

Deep contextualized embeddings trained using unsupervised language modeling (e.g., ELMo and BERT) are successful in a wide range of NLP tasks. In this paper, we propose a new contextualized embedding model of words and entities for named entity disambiguation (NED). Our model is based on the bidirectional transformer encoder and produces contextualized embeddings for words and entities in the input text. The embeddings are trained using a new *masked entity prediction* task that aims to train the model by predicting randomly masked entities in entity-annotated texts. We trained the model using entity-annotated texts obtained from Wikipedia. We evaluated our model by addressing NED using a simple NED model based on the trained contextualized embeddings. As a result, we achieved state-of-the-art or competitive results on several standard NED datasets.

1 Introduction

Named entity disambiguation (NED) refers to the task of assigning entity mentions in a text to corresponding entries in a knowledge base (KB). This task is challenging owing to the ambiguity between entity names (e.g., "World Cup") and the entities they refer to (e.g., `FIFA World Cup` and `Rugby World Cup`).

Deep contextualized word embedding models, e.g., ELMo (Devlin et al., 2018) and BERT (Devlin et al., 2018), have recently achieved state-of-the-art results on many tasks. Unlike conventional word embedding models that assign a single, fixed embedding per word, these models produce a contextualized embedding for each word in the input text using a pretrained neural network encoder. The encoder can be a recurrent neural network or transformer (Vaswani et al., 2017), and is usually trained using an unsupervised objective based on language modeling. For instance, Devlin et al. (2018) proposed Masked Language Model (MLM), which aims to train the embeddings by predicting randomly masked words in the text.

In this paper, we describe a new contextualized embedding model for words and entities for NED. Following Devlin et al. (2018), the proposed model is based on the bidirectional transformer encoder (Vaswani et al., 2017). It takes a sequence of words and entities in the input text, and produces a contextualized embedding for each word and entity. Inspired by MLM, we propose *masked entity prediction*, a new task that aims to train the embedding model by predicting randomly masked entities based on words and non-masked entities in the input text. We trained the model using texts and their entity annotations retrieved from Wikipedia.

We evaluated the proposed model by addressing NED using an NED model based on trained contextualized embeddings. The NED model addresses the task by capturing word-based and entity-based contextual information using the trained contextualized embeddings. As a result, we achieved state-of-the-art or competitive results on various standard NED datasets. We will release our code and trained embeddings for further research.

2 Background and Related Work

Neural network-based approaches have recently achieved strong results on NED (Ganea and Hofmann, 2017; Yamada et al., 2017; Eshel et al., 2017; Le and Titov, 2018; Cao et al., 2018). A key component of these approaches is an embedding model of words and entities trained using a large knowledge base (e.g., Wikipedia). Such embedding models enable us to design NED models that capture the contextual information required to address NED. These models are typically based on conventional word embedding models (e.g., skipgram (Mikolov et al., 2013)) that assign a fixed embedding to each word and entity (Yamada et al., 2016; Fang et al., 2016; Tsai and Roth, 2016; Yamada et al., 2017; Cao et al., 2017; Ganea and Hofmann, 2017). In this study, we aim to test the effectiveness of the pretrained contextualized embeddings for NED.

3 Contextualized Embeddings of Words and Entities

In this section, we introduce our contextualized embedding model for words and entities. Figure 1 shows the architecture of the proposed model. Our model adopts a multi-layer bidirectional transformer encoder (Vaswani et al., 2017)[1] with input representations described later in this section. Given a sequence of tokens consisting of words and entities, the model first represents the sequence as a sequence of input embeddings, one for each token, and then the model generates a contextualized output embedding for each token. Both input and output embeddings have H dimensions. Hereafter, we denote the number of words and that of entities in the vocabulary of our model by V_w and V_e, respectively.

3.1 Input Representation

Similar to the approach adopted in Devlin et al. (2018), the input representation of a given token (i.e., word or entity) is constructed by summing the following three embeddings of H dimensions:

- **Token embedding** is the embedding of the corresponding token. The matrices of the word and entity token embeddings are represented as $\mathbf{A} \in \mathbb{R}^{V_w \times H}$ and $\mathbf{B} \in \mathbb{R}^{V_e \times H}$, respectively.
- **Token type embedding** represents the type of token, namely word type (denoted by $\mathbf{C}_{word}$) or entity type (denoted by $\mathbf{C}_{entity}$).
- **Position embedding** represents the position of the token in a word sequence. A word and an entity appearing at i-th position in the sequence are represented as $\mathbf{D}_i$ and $\mathbf{E}_i$, respectively. If an entity name contains multiple words, we compute its position embedding by averaging the embeddings of the corresponding positions (e.g., `New York City` in Figure 1).

Following Devlin et al. (2018), we insert special word tokens `[CLS]` and `[SEP]` to the word sequence as the first and last words, respectively.

3.2 Masked Entity Prediction

To train the embeddings, we propose *masked entity prediction* (MEP), a new task based on MLM. In particular, we mask some percentage of the input entities at random; then, we train the embeddings to predict masked entities based on words and non-masked entities. We represent masked entities using the special `[MASK]` entity token.

We adopt a model equivalent to the one used to predict words in MLM. Specifically, we predict the original entity of a masked entity by applying the softmax function over all entities in our vocabulary:

$$\hat{\mathbf{y}}_{MEP} = \text{softmax}(\mathbf{B}\mathbf{m} + \mathbf{b}_o), \tag{1}$$

where $\mathbf{b}_o \in \mathbb{R}^{V_e}$ is the output bias, and $\mathbf{m} \in \mathbb{R}^H$ is derived as:

$$\mathbf{m} = \text{layer\_norm}\big(\text{gelu}(\mathbf{W}_f\mathbf{h} + \mathbf{b}_f)\big), \tag{2}$$

where $\mathbf{h} \in \mathbb{R}^H$ is the output embedding corresponding to the masked entity, $\mathbf{W}_f \in \mathbb{R}^{H \times H}$ is weight matrix, $\mathbf{b}_f \in \mathbb{R}^H$ is the bias, gelu$(\cdot)$ is the gelu activation function (Hendrycks and Gimpel, 2016), and layer_norm$(\cdot)$ is the layer normalization function (Lei Ba et al., 2016).

3.3 Training

We used the same model configuration adopted in the BERT$_{\text{LARGE}}$ model (Devlin et al., 2018). In particular, we used the bidirectional transformer encoder with $H = 1024$ hidden dimensions, 24 hidden layers, 16 self-attention heads, and the gelu activation function (Hendrycks and Gimpel, 2016). We also set the feed-forward/filter size to 4096, the dropout probability applied to all layers was 0.1, and the maximum word length in an input sequence was set to 512. Furthermore, we initialized the parameters of our model that were common with BERT (i.e., parameters in the transformer encoder and the embeddings for words) using the uncased version of the pretrained BERT$_{\text{LARGE}}$ model.[2] Other parameters, namely the parameters in the MEP and the embeddings for entities, were initialized randomly.

The model was trained via iterations over Wikipedia pages in a random order for two epochs.

[1] For details of the transformer encoder, refer to Vaswani et al. (2017).

[2] We initialized $\mathbf{C}_{word}$ using BERT's segment embedding for sentence A.

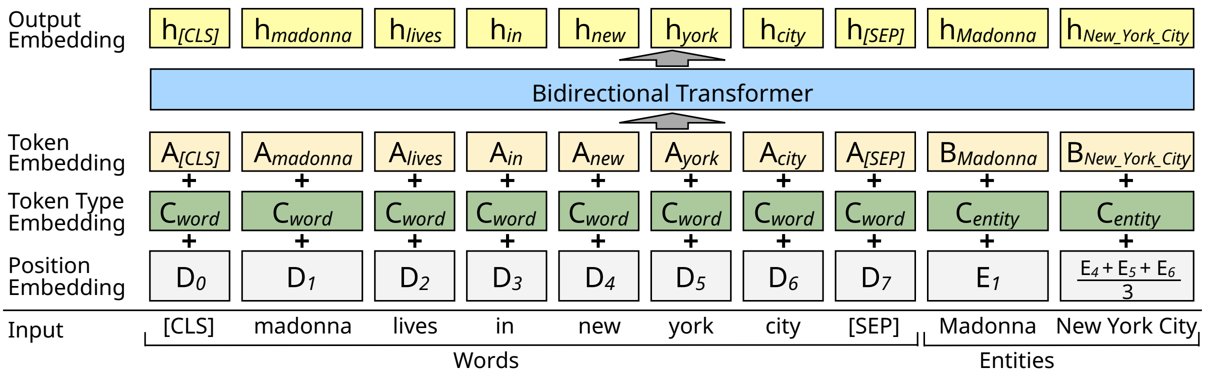

Figure 1: Architecture of the proposed contextualized embedding model of words and entities.

We generated input sequences by splitting the content of each page into sequences consisting of ≤ 512 words and their entity annotations (i.e., hyperlinks). We used the December 2018 version of Wikipedia, consisting of approximately 3.5 billion words and 11 million entity annotations. We masked 30% of all entities in each sequence at random. The input text was lowercased and tokenized to words[3] using the BERTs sub-word tokenizer (Devlin et al., 2018) with its vocabulary consisting of $V_w = 30,000$ words. Similar to Ganea and Hofmann (2017), we built an entity vocabulary consisting of $V_e = 221,965$ entities that were contained in the entity candidates in the NED datasets described in Section 4.1.

We used the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 2e-5, $\beta 1 = 0.9$, $\beta 2 = 0.999$, and L2 weight decay of 0.01. The batch size was set to 252. We trained the model by maximizing the log likelihood of MEP's predictions. To stabilize training, we updated only parameters that were initialized randomly (i.e., fixed the parameters initialized using BERT) at the first epoch, and updated all parameters at the second epoch. The training took approximately six days using eight Tesla V100 GPUs.

4 NED Based on Pre-trained Contextualized Embeddings

In this section, we address NED based on the proposed contextualized embeddings.

4.1 Experimental Setup

Our experimental setup described in this section follows past work (Ganea and Hofmann, 2017; Le and Titov, 2018). In particular, we test the NED models using *in-domain* and *out-domain* scenarios. In the in-domain scenario, we use the *train* and *test_b* sets of the AIDA-CoNLL dataset (Hoffart et al., 2011) to train and test the models, respectively. In the out-domain scenario, we test the generalization ability of the models on five test sets: MSNBC (MSB), AQUAINT (AQ), ACE2004 (ACE), which were cleaned by Guo and Barbosa (2018), WNED-CWEB (CWEB), and WNED-WIKI (WW), which were obtained from ClueWeb and Wikipedia (Guo and Barbosa, 2018; Gabrilovich et al., 2013). For all datasets in both scenarios, we use the standard *KB+YAGO* entity candidates and their associated prior probabilities ($\hat{p}(e|m)$) (Ganea and Hofmann, 2017), and use only the top 30 candidates based on $\hat{p}(e|m)$. We consider only mentions that refer to valid entities in Wikipedia. We report the accuracy for the in-domain scenario, and the micro F1 score (averaged per mention) for the out-domain scenario.

4.2 Model Inputs

For each mention in the input document, we create an input sequence consisting of (1) a masked entity corresponding to the mention, (2) words in the document[4], and optionally (3) entities obtained from pseudo entity annotations.

Pseudo entity annotations are created by treating all mentions except the target mention in the document as entity annotations referring to their entity candidates. For each mention, we create a pseudo entity annotation for each entity candidate of the mention. For efficiency, we ignore a candidate if its $\hat{p}(e|m)$ is less than 0.1. Note that the contextual information obtained from entities

[3]In this paper, a word can refer to a linguistic word and a sub-word.

[4]If the document is longer than the maximum word length (i.e., 512 words), we remove words in the order of greatest distance from the target mention.

appearing in the same document has been considered as critical in improving NED and a main focus of the past literature on NED. Because the transformer encoder is based on a neural attention mechanism to compute the embedding of a token by automatically attending relevant tokens in the input document, we assume that the trained transformer encoder can selectively attend to relevant entities if we input noisy and likely irrelevant entities based on the pseudo annotations.

4.3 Model

Our NED model is based on our pre-trained contextualized embeddings. For each entity mention with its K entity candidates, our NED model first takes the input sequence described above, and computes the vector $\mathbf{m}' \in \mathbb{R}^H$ corresponding to the mention using Eq. (2). Then, the model predicts the referent entity using the softmax function over the entity candidates:

$$\hat{\mathbf{y}}_{NED} = \text{softmax}(\mathbf{B}^*\mathbf{m}' + \mathbf{b}_o^*),$$

where $\mathbf{B}^* \in \mathbb{R}^{K\times H}$ and $\mathbf{b}_o^* \in \mathbb{R}^K$ consists of the entity token embeddings and the output bias values corresponding to the entity candidates, respectively. Note that $\mathbf{B}^*$ and $\mathbf{b}_o^*$ are the subsets of $\mathbf{B}$ and $\mathbf{b}_o$, respectively.

In the in-domain scenario, we fine-tuned the model by maximizing the log likelihood of the NED predictions on the training set of the AIDA-CoNLL dataset. During the training, we fixed the entity token embeddings ($\mathbf{B}$ and $\mathbf{B}^*$) and output bias ($\mathbf{b}_o$ and $\mathbf{b}_o^*$), and updated all other parameters. We set the batch size to 32, and used the Adam optimizer with a learning rate of 2e-5, $\beta 1 = 0.9$, $\beta 2 = 0.999$, and L2 weight decay of 0.01. The training consisted of two epochs.[5] In the out-domain scenario, we did not perform fune-tuning.

4.4 Results

The results of the in-domain scenario are shown in Table 1.[6] Several models, including our models, report the 95% confidence intervals obtained over five runs. As shown, our models outperformed all previously proposed models. Furthermore, using pseudo entity annotations boosted the accuracy by 0.3%.

[5]We tuned the number of epochs, batch size, and learning rate using the development set of the AIDA-CoNLL dataset.

[6]For fair comparison and following past work (Ganea and Hofmann, 2017; Le and Titov, 2018), we do not compare our models against models based on PPRforNED entity candidates (Pershina et al., 2015).

Methods	Accuracy
Chisholm and Hachey (2015)	88.7
Guo and Barbosa (2018)	89.0
Globerson et al. (2016)	91.0
Yamada et al. (2016)	91.5
Ganea and Hofmann (2017)	92.22 ± 0.14
Yang et al. (2018)	93.0
Le and Titov (2018)	93.07 ± 0.27
Our	94.0 ± 0.28
Our (+pseudo entities)	94.3 ± 0.25

Table 1: Accuracies on the AIDA-CoNLL dataset.

Methods	MSB	AQ	ACE	CWEB	WW
Milne and Witten (2008)	78	85	81	64.1	81.7
Hoffart et al. (2011)	79	56	80	58.6	63
Ratinov et al. (2011)	75	83	82	56.2	67.2
Cheng and Roth (2013)	90	90	86	67.5	73.4
Ganea and Hofmann (2017)	93.7	88.5	88.5	77.9	77.5
Guo and Barbosa (2018)	92	87	88	77	84.5
Yang et al. (2018)	92.6	90.5	89.2	**81.8**	79.2
Le and Titov (2018)	93.9	88.3	89.9	77.5	78.0
Cao et al. (2018)	-	87	88	-	86
Our	**94.1**	92.5	91.1	76.2	**86.2**
Our (+pseudo entities)	93.8	**93.7**	**91.5**	74.9	83.9

Table 2: Micro F1 scores for the datasets used in our out-domain scenario.

The results of the out-domain scenario are shown in Table 2. Our models achieved new state-of-the-art results on four of the five datasets, namely MSNBC, AQUAINT, ACE2004, and WNED-WIKI, and performed competitive on the WNED-CLUEWEB dataset. Furthermore, using pseudo entity annotations improved the performance on the AQUAINT and ACE2004 datasets.

Note that unlike all the past models shown in Table 2 except Cao et al. (2018), our models used in the out-domain scenario were trained only on entity annotations retrieved from Wikipedia. Therefore, our models can be easily applied to any languages in which Wikipedia is available.

5 Conclusions

In this work, we proposed a contextualized embedding model of words and entities for NED. We also introduced MEP to train the model using entity-annotated texts as inputs. We trained the model using Wikipedia and evaluated the effectiveness of the model by addressing NED using various standard NED datasets. The experimental results show the competitiveness of our model across a wide range of NED datasets. In the future, we intend to improve our NED model by modeling the global coherence of the disambiguated entities.

References

Yixin Cao, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2018. Neural Collective Entity Linking. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 675–686.

Yixin Cao, Lifu Huang, Heng Ji, Xu Chen, and Juanzi Li. 2017. Bridge Text and Knowledge by Learning Multi-Prototype Entity Mention Embedding. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1623–1633.

Xiao Cheng and Dan Roth. 2013. Relational Inference for Wikification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1787–1796.

Andrew Chisholm and Ben Hachey. 2015. Entity Disambiguation with Web Links. *Transactions of the Association for Computational Linguistics*, 3:145–156.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*.

Yotam Eshel, Noam Cohen, Kira Radinsky, Shaul Markovitch, Ikuya Yamada, and Omer Levy. 2017. Named Entity Disambiguation for Noisy Text. In *Proceedings of the 21st Conference on Computational Natural Language Learning*, pages 58–68.

Wei Fang, Jianwen Zhang, Dilin Wang, Zheng Chen, and Ming Li. 2016. Entity Disambiguation by Knowledge and Text Jointly Embedding. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 260–269.

Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. 2013. FACC1: Freebase Annotation of ClueWeb Corpora, Version 1 (Release date 2013-06-26, Format version 1, Correction level 0).

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep Joint Entity Disambiguation with Local Neural Attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629.

Amir Globerson, Nevena Lazic, Soumen Chakrabarti, Amarnag Subramanya, Michael Ringaard, and Fernando Pereira. 2016. Collective Entity Resolution with Multi-Focal Attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 621–631.

Zhaochen Guo and Denilson Barbosa. 2018. Robust Named Entity Disambiguation with Random Walks. *Semantic Web*, 9(4):459–479.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian Error Linear Units (GELUs). *arXiv preprint arXiv:1606.08415*.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust Disambiguation of Named Entities in Text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792.

Diederik Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980v9*.

Phong Le and Ivan Titov. 2018. Improving Entity Linking by Modeling Latent Relations between Mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1595–1604.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer Normalization. *arXiv preprint arXiv:1607.06450*.

Tomas Mikolov, Greg Corrado, Kai Chen, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of the 2013 International Conference on Learning Representations*, pages 1–12.

David Milne and Ian H. Witten. 2008. Learning to Link with Wikipedia. In *Proceeding of the 17th ACM Conference on Information and Knowledge Management*, pages 509–518.

Maria Pershina, Yifan He, and Ralph Grishman. 2015. Personalized Page Rank for Named Entity Disambiguation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 238–243.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and Global Algorithms for Disambiguation to Wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1375–1384.

Chen-Tse Tsai and Dan Roth. 2016. Cross-lingual Wikification Using Multilingual Embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 589–598.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 250–259.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2017. Learning Distributed Representations of Texts and Entities from Knowledge Base. *Transactions of the Association for Computational Linguistics*, 5:397–411.

Yi Yang, Ozan Irsoy, and Kazi Shefaet Rahman. 2018. Collective Entity Disambiguation with Structured Gradient Tree Boosting. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 777–786.