Recent Advances in Pre-training Methods for NLP

Pei Zhou
Department of Computer Science
University of Southern California
peiz@usc.edu

Abstract

Pre-trained language representation models, such as BERT, capture a general language representation from large-scale corpora. Recent advances include augmenting language models (LM) with structured knowledge to make better predictions in knowledge driven or domain-specific tasks. This lecture will briefly introduce most recent papers on this aspect.

1 Introduction

Since the year of 2017, pre-trained LMs have made some of the most exciting breakthroughs in the field of Nartual Language Processing (NLP) and numerous new models are designed to tackle different tasks. As Figure 1 shows, starting from ELMo Peters et al. [2018] and BERT Devlin et al. [2018], people have tried incorporating cross-lingual training, multi-tasking, knowledge graphs, images and videos into LMs. This lecture mainly focuses on recent models combining LMs with structured knowledge.

2 Knowledge Enhanced Language Representations

2.1 Motivation

Self-supervised learning of LMs often neglect structured knowledge information, making knowledge-driven tasks like entity typing and relation extraction hard. Knowledge graphs (KGs) are argued to be able to provide rich structured knowledge facts for better language understanding. There are two main challenges according to Zhang et al. [2019]. (1) **Structured Knowledge Encoding**: regarding to the given text, how to effectively extract and encode its related informative facts in KGs for language representation models is an important problem; (2) **Heterogeneous Information Fusion**: the pre-training procedure for language representation is quite different from the knowledge representation procedure, leading to two individual vector spaces.

There has been work that tries to integrate language modeling with knowledge graphs. Specifically, we are going to focus on two ERNIEs proposed by THU Zhang et al. [2019] and Baidu Sun et al. [2019], and K-BERT Liu et al. [2019].

2.2 Models

2.2.1 THU-ERNIE

Overview They propose Enhanced Language RepresentatioN with Informative Entities (ERNIE), which pre-trains a language representation model on both large-scale textual corpora and KGs: (1) For extracting and encoding knowledge information, they first recognize named entity mentions in text and then align these mentions to their corresponding entities in KGs. Instead of directly using the graph-based facts in KGs, they encode the graph structure of KGs with knowledge embedding

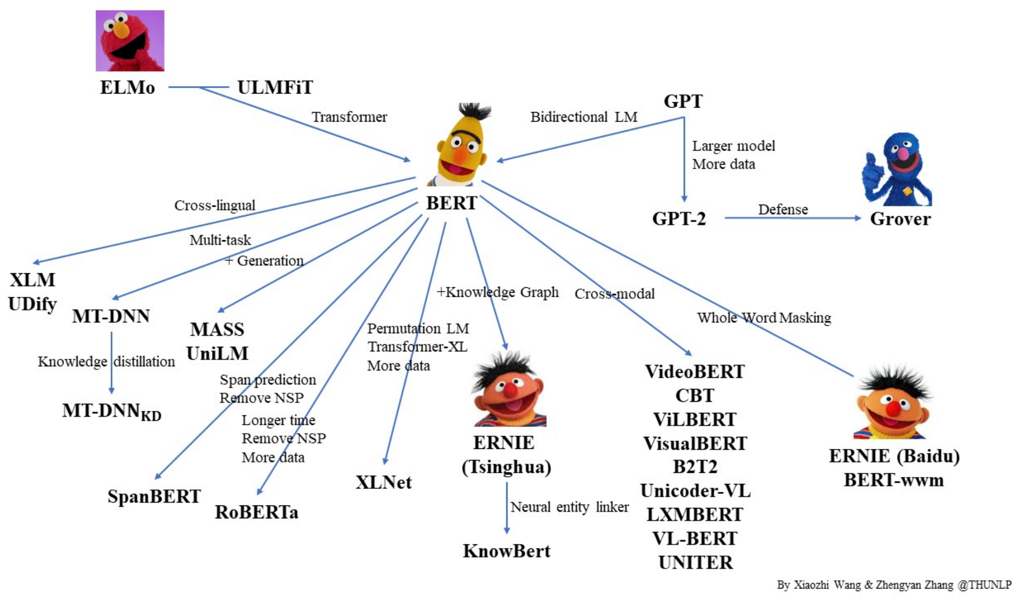


Figure 1: A diagram designed by students from Tsinghua University with a github repo of curated paper list on LMs: `https://github.com/thunlp/PLMpapers`

algorithms like TransE Bordes et al. [2013], and then take the informative entity embeddings as input for ERNIE. Based on the alignments between text and KGs, ERNIE integrates entity representations in the knowledge module into the underlying layers of the semantic module.

(2) Similar to BERT, they adopt the masked language model and the next sentence prediction as the pre-training objectives. Besides, for the better fusion of textual and knowledge features, they design a new pre-training objective by randomly masking some of the named entity alignments in the input text and asking the model to select appropriate entities from KGs to complete the alignments. Unlike the existing pre-trained language representation models only utilizing local context to predict tokens, their objectives require models to aggregate both context and knowledge facts for predicting both tokens and entities, and lead to a knowledgeable language representation model.

Architecture The whole model architecture of ERNIE consists of two stacked modules as seen in Figure 2: (1) the underlying textual encoder (T-Encoder) responsible to capture basic lexical and syntactic information from the input tokens, and (2) the upper knowledgeable encoder (K-Encoder) responsible to integrate extra token-oriented knowledge information into textual information from the underlying layer, so that they can represent heterogeneous information of tokens and entities into a united feature space. The T-Encoder is simply a multi-layer bidirectional Transformer encoder identical to BERT.

Knowledgeable Encoder The knowledgeable encoder K-Encoder consists of stacked aggregators, which are designed for encoding both tokens and entities as well as fusing their heterogeneous features. In the i-th aggregator, the input token embeddings $\left\{\mathbf{w}_1^{(i-1)}, ..., \mathbf{w}_n^{(i-1)}\right\}$ and entity embeddings $\left\{\mathbf{e}_1^{(i-1)}, ..., \mathbf{e}_m^{(i-1)}\right\}$ from the preceding aggregator are fed into two multi-head self-attentions (MH-ATTs) Vaswani et al. [2017], respectively,

$$\left\{\tilde{\mathbf{w}}_1^{(i)}, ..., \tilde{\mathbf{w}}_n^{(i)}\right\} = \texttt{MH-ATT}(\left\{\mathbf{w}_1^{(i-1)}, ..., \mathbf{w}_n^{(i-1)}\right\})$$
$$\left\{\tilde{\mathbf{e}}_1^{(i)}, ..., \tilde{\mathbf{e}}_m^{(i)}\right\} = \texttt{MH-ATT}(\left\{\mathbf{e}_1^{(i-1)}, ..., \mathbf{e}_m^{(i-1)}\right\})$$

Then, the i-th aggregator adopts an information fusion layer for the mutual integration of the token and entity sequence, and computes the output embedding for each token and entity. For a token w_j

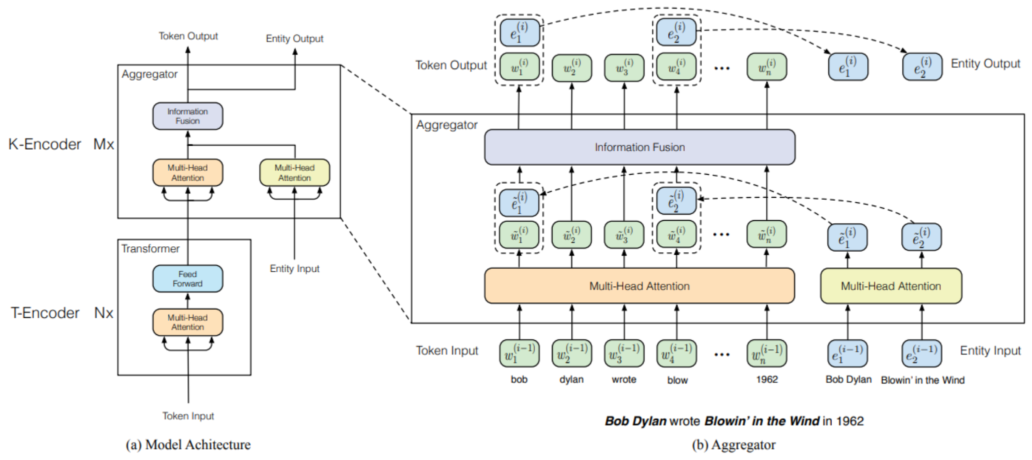


Figure 2: The left part is the architecture of ERNIE. The right part is the aggregator for the mutual integration of the input of tokens and entities.

and its aligned entity $e_k = f(w_j)$, the information fusion process is as follows,

$$\begin{aligned}
\mathbf{h}_j &= \sigma(\tilde{\mathbf{W}}_t^{(i)}\tilde{\mathbf{w}}_j^{(i)} + \tilde{\mathbf{W}}_e^{(i)}\tilde{\mathbf{e}}_k^{(i)} + \tilde{\mathbf{b}}^{(i)}) \\
\mathbf{w}_j^{(i)} &= \sigma(\mathbf{W}_t^{(i)}\mathbf{h}_j + \mathbf{b}_t^{(i)}) \\
\mathbf{e}_k^{(i)} &= \sigma(\mathbf{W}_e^{(i)}\mathbf{h}_j + \mathbf{b}_e^{(i)})
\end{aligned}$$

where $\mathbf{h}_j$ is the inner hidden state integrating the information of both the token and the entity. $\sigma(\cdot)$ is the non-linear activation function, which usually is the GELU function Hendrycks and Gimpel [2016]. For the tokens without corresponding entities, the information fusion layer computes the output embeddings without integration.

The output embeddings of both tokens and entities computed by the top aggregator will be used as the final output embeddings of the knowledgeable encoder.

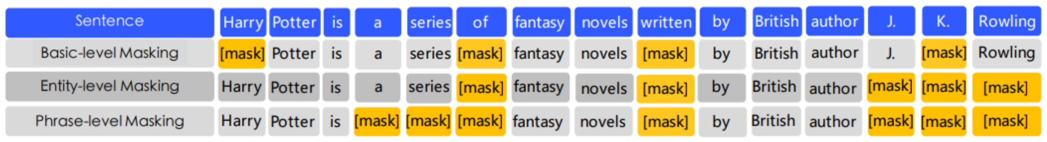
New Pre-training Objective In order to inject knowledge into language representation by informative entities, they propose a new pre-training task for ERNIE, which randomly masks some token-entity alignments and then requires the system to predict all corresponding entities based on aligned tokens. Given the token sequence $w_1, ..., w_n$ and its corresponding entity sequence $e_1, ..., e_m$, they define the aligned entity distribution for the token w_i as follows,

$$p(e_j|w_i) = \frac{\exp(\texttt{linear}(\mathbf{w}_i^o) \cdot \mathbf{e}_j)}{\sum_m^{k=1} \exp(\texttt{linear}(\mathbf{w}_i^o) \cdot \mathbf{e}_k)},$$

which will be used to compute the cross-entropy loss function.

Considering that there are some errors in token-entity alignments, they perform the following operations: (1) In 5% of the time, for a given token-entity alignment, they replace the entity with another random entity, which aims to train the model to correct the errors that the token is aligned with a wrong entity; (2) In 15% of the time, they mask token-entity alignments, which aims to train their model to correct the errors that the entity alignment system does not extract all existing alignments; (3) In the rest of the time, they keep token-entity alignments unchanged, which aims to encourage the model to integrate the entity information into token representations for better language understanding.

Side note: a new paper about a model KnowBert from authors of ELMo is accepted at EMNLP this year and adopts a similar idea of integrating Knowledge Bases (KBs) to LMs Peters et al. [2019].



Sentence	Harry	Potter	is	a	series	of	fantasy	novels	written	by	British	author	J.	K.	Rowling
Basic-level Masking	[mask]	Potter	is	a	series	[mask]	fantasy	novels	[mask]	by	British	author	J.	[mask]	Rowling
Entity-level Masking	Harry	Potter	is	a	series	[mask]	fantasy	novels	[mask]	by	British	author	[mask]	[mask]	[mask]
Phrase-level Masking	Harry	Potter	is	[mask]	[mask]	[mask]	fantasy	novels	[mask]	by	British	author	[mask]	[mask]	[mask]

Figure 3: Different masking level of a sentence

2.2.2 Baidu-ERNIE

Overview In addition to basic masking strategy, they use two kinds of knowledge strategies: phrase-level strategy and entity-level strategy. They take a phrase or a entity as one unit, which is usually composed of several words. All of the words in the same unit are masked during word representation training, instead of only one word or character being masked. In this way, the prior knowledge of phrases and entities are implicitly learned during the training procedure. Instead of adding the knowledge embedding directly, ERNIE implicitly learned the information about knowledge and longer semantic dependency, such as the relationship between entities, the property of a entity and the type of a event, to guide word embedding learning. This can make the model have better generalization and adaptability.

In order to reduce the training cost of the model, ERNIE is pre-trained on heterogeneous Chinese data, and then applied to 5 Chinese NLP tasks. ERNIE advances the state-of-the-art results on all of these tasks. An additional experiment on the cloze test shows that ERNIE has better knowledge inference capacity over other strong baseline methods.

Knowledge Integration Masking Instead of adding the knowledge embedding directly, they proposed a multi-stage knowledge masking strategy to integrate phrase and entity level knowledge into the language representation. The different masking level of a sentence is described in Figure 3.

The first stage is the same as BERT. The second stage is to employ phrase-level masking. Phrase is a small group of words or characters together acting as a conceptual unit. For English, they use lexical analysis and chunking tools to get the boundary of phrases in the sentences, and use some language dependent segmentation tools to get the word/phrase information in other language such as Chinese. In phrase-level mask stage, they also use basic language units as training input, unlike random basic units mask, this time they randomly select a few phrases in the sentence, mask and predict all the basic units in the same phrase. At this stage, phrase information is encoded into the word embedding.

The third stage is entity-level masking. Name entities contain persons, locations, organizations, products, etc., which can be denoted with a proper name. It can be abstract or have a physical existence. Usually entities contain important information in the sentences. As in the phrase masking stage, they first analyze the named entities in a sentence, and then mask and predict all slots in the entities. After three stages, word representation enhanced by richer semantic information is obtained.

2.2.3 K-BERT

Overview They propose a knowledge-enabled language representation model (K-BERT) with knowledge graphs (KGs), in which triples are injected into the sentences as domain knowledge. However, too much knowledge incorporation may divert the sentence from its correct meaning, which is called knowledge noise (KN) issue. To overcome KN, K-BERT introduces soft-position and visible matrix to limit the impact of knowledge. K-BERT can easily inject domain knowledge into the models by equipped with a KG without pre-training by-self because it is capable of loading model parameters from the pretrained BERT.

Architecture As shown in Figure 4, the model architecture of K-BERT consists of four modules, i.e., knowledge layer, embedding layer, seeing layer and mask-transformer. For an input sentence, the knowledge layer first injects relevant triples into it from a KG, transforming the original sentence into a knowledge-rich sentence tree. The sentence tree is then simultaneously fed into the embedding layer and the seeing layer and then converted to a token-level embedding representation and a visible

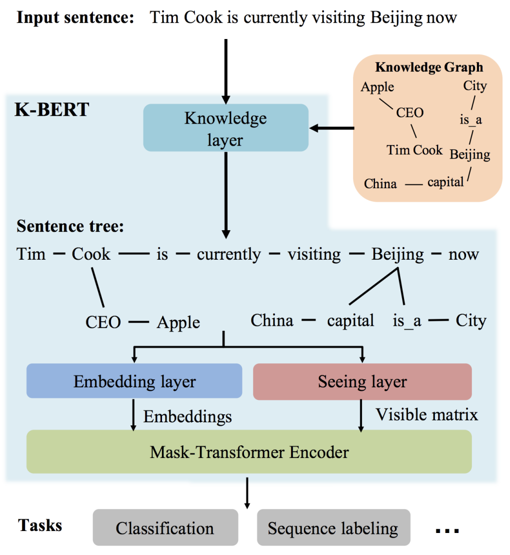


Figure 4: The model structure of K-BERT: Compared to other RL models, the K-BERT is equipped with an editable KG, which can be adapted to its application domain.

matrix. The visible matrix is used to control the visible area of each token, preventing changing the meaning of the original sentence due to too much knowledge injected.

Knowledge Layer The knowledge layer (KL) is used for sentence knowledge injection and sentence tree conversion. Specifically, given an input sentence $s = \{w_0, w_1, ..., w_n\}$ and a KG $\mathbb{K}$, KL outputs a sentence tree $t = \{w_0, w_1, ..., w_i \{(r_{i0}, w_{i0}), ..., (r_{ik}, w_{ik})\}, ..., w_n\}$. In this work, a sentence tree can have multiple branches, but its depth is fixed to 1, which means that the entity names in triples will not derive branches iteratively.

Soft-position Embedding Taking the sentence tree in Figure 5 as an example, after rearranging, [CEO] and [Apple] are inserted between [Cook] and [is], but the subject of [is] should be [Cook] instead of [Apple]. To solve this problem, they set the position number of [is] to 3 instead of 5. So when calculating the self-attention score in the transformer encoder, [is] is at the next position of [Cook] by the equivalent. However, another problem arises: the position numbers of [is] and [CEO] are both 3, which makes them close in position when calculating self-attention, but in fact, there is no connection between them. The solution to this problem is Mask-Self-Attention, which will be covered in the next subsection.

Seeing Layer The input to K-BERT is a sentence tree, where the branch is the knowledge gained from KG. However, the risk raised with knowledge is that it can lead to changes in the meaning of the original sentence, i.e., KN issue. For example, in the sentence tree in Figure 2, [China] only modifies [Beijing] and has nothing to do with [Apple]. Therefore, the representation of [Apple] should not be affected by [China]. On the other hand, the [CLS] tag used for classification should not bypass the [Cook] to get the information of [Apple], as this would bring the risk of semantic changes. To prevent this from happening, K-BERT's use a visible matrix M to limit the visible area of each token so that [Apple] and [China], [CLS] and [Apple] are not visible to each other. The visible matrix M is defined as,

$$M_{ij} = \begin{cases} 0 & w_i \ominus w_j \\ -\infty & w_i \oslash w_j \end{cases}$$

where $w_i \ominus w_j$ indicates that w_i and w_j are in the same branch, while $w_i \oslash w_j$ are not. i and j are the hard-position index.

Mask-Self-Attention The Transformer Vaswani et al. [2017] encoder in BERT cannot receive the visible matrix M as an input, so they need to modify it to Mask-Transformer, which can limit the

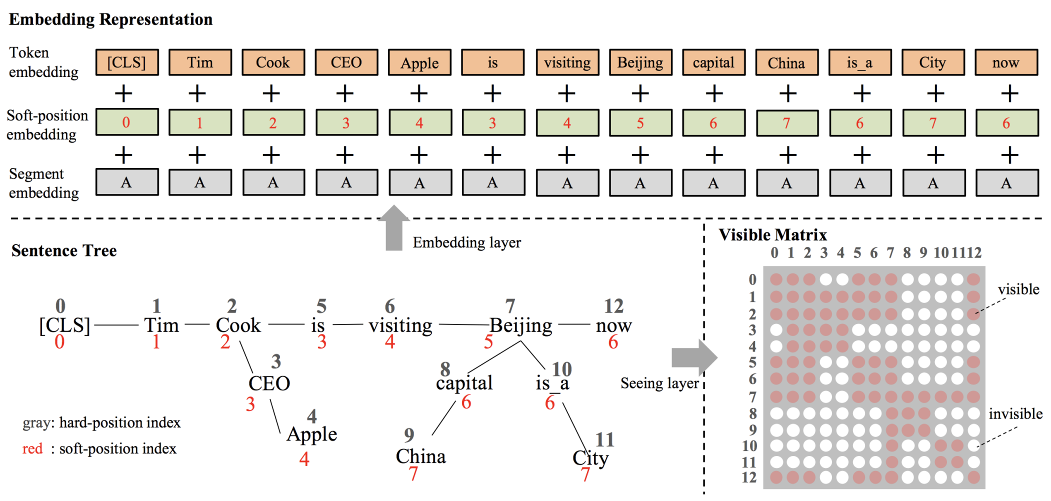


Figure 5: The process of converting a sentence tree into an embedding representation and a visible matrix. In the visible matrix, red means visible, and white means invisible. For example, the cell at row 4, column 9 is white means that the "Apple(4)" cannot see "China(9)".

self-attention region according to M. Mask-Transformer is a stack of multiple mask-self-attention blocks. Formally, the mask-self-attention is,

$$
\begin{aligned}
Q^{i+1}, K^{i+1}, V^{i+1} &= h^i W_q, h^i W_k, h^i W_v, \\
S^{i+1} &= \text{softmax}\left(\frac{Q^{i+1} {K^{i+1}}^{\top} + M}{\sqrt{d_k}}\right), \\
h^{i+1} &= S^{i+1} V^{i+1},
\end{aligned}
$$

where W_q, W_k and W_v are trainable model parameters. h_i is the hidden state of the i-th mask-self-attention blocks. d_k is the scaling factor. M is the visible matrix calculated by the seeing layer. Intuitively, if w_k is invisible to w_j , the M_{jk} will mask the attention score S^{i+1}_{jk} to 0, which means w_k make no contribution to the hidden state of w_j.

References

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

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

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. Ernie: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129*, 2019.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*, 2019.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. K-bert: Enabling language representation with knowledge graph. *arXiv preprint arXiv:1909.07606*, 2019.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Matthew E Peters, Mark Neumann, IV Logan, L Robert, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. Knowledge enhanced contextual word representations. *arXiv preprint arXiv:1909.04164*, 2019.