

Enhancing Pre-Trained Language Representations with Rich Knowledge for Machine Reading Comprehension

An Yang^{1*}, Quan Wang², Jing Liu², Kai Liu²,
Yajuan Lyu², Hua Wu^{2†}, Qiaoqiao She² and Sujian Li^{1†}

¹Key Laboratory of Computational Linguistics, Peking University, MOE, China

²Baidu Inc., Beijing, China

{yangan, lisujian}@pku.edu.cn {wangquan05, liujing46,
liukai20, lvayajuan, wuhua, sheqiaoqiao}@baidu.com

Abstract

Machine reading comprehension (MRC) is a crucial and challenging task in NLP. Recently, pre-trained language models (LMs), especially BERT, have achieved remarkable success, presenting new state-of-the-art results in MRC. In this work, we investigate the potential of leveraging external knowledge bases (KBs) to further improve BERT for MRC. We introduce KT-NET, which employs an attention mechanism to adaptively select desired knowledge from KBs, and then fuses selected knowledge with BERT to enable context- and knowledge-aware predictions. We believe this would combine the merits of both deep LMs and curated KBs towards better MRC. Experimental results indicate that KT-NET offers significant and consistent improvements over BERT, outperforming competitive baselines on ReCoRD and SQuAD1.1 benchmarks. Notably, it ranks the 1st place on the ReCoRD leaderboard, and is also the best single model on the SQuAD1.1 leaderboard at the time of submission (March 4th, 2019).¹

1 Introduction

Machine reading comprehension (MRC), which requires machines to comprehend text and answer questions about it, is a crucial task in natural language processing. With the development of deep learning and the increasing availability of datasets (Rajpurkar et al., 2016, 2018; Nguyen et al., 2016; Joshi et al., 2017), MRC has achieved remarkable advancements in the last few years.

Recently language model (LM) pre-training has caused a stir in the MRC community. These LMs

^{*}This work was done while the first author was an intern at Baidu Inc.

[†]Co-corresponding authors: Hua Wu and Sujian Li.

¹Our code will be available at <http://github.com/paddlepaddle/models/tree/develop/PaddleNLP/Research/ACL2019-KTNET>

Passage: The US government has extended its review into whether trade sanctions against Sudan should be repealed. [...] Sudan is committed to the full implementation of UN Security Council resolutions on North Korea. [...] Sudan's past support for North Korea could present an obstacle [...]

Question: Sudan remains a XXX-designated state sponsor of terror and is one of six countries subject to the Trump administration's ban.

Original BERT prediction: UN Security Council

Prediction with background knowledge: US

Background knowledge:

NELL: (Donald Trump, person-leads-organization, US)

WordNet: (government, same-synset-with, administration)

WordNet: (sanctions, common-hyponym-with, ban)

Figure 1: An example from ReCoRD, with answer candidates marked (underlined) in the passage. The vanilla BERT model fails to predict the correct answer. But it succeeds after integrating background knowledge collected from WordNet and NELL.

are pre-trained on unlabeled text and then applied to MRC, in either a feature-based (Peters et al., 2018a) or a fine-tuning (Radford et al., 2018) manner, both offering substantial performance boosts. Among different pre-training mechanisms, BERT (Devlin et al., 2018), which uses Transformer encoder (Vaswani et al., 2017) and trains a bidirectional LM, is undoubtedly the most successful by far, presenting new state-of-the-art results in MRC and a wide variety of other language understanding tasks. Owing to the large amounts of unlabeled data and the sufficiently deep architectures used during pre-training, advanced LMs such as BERT are able to capture complex linguistic phenomena, understanding language better than previously appreciated (Peters et al., 2018b; Goldberg, 2019).

However, as widely recognized, genuine reading comprehension requires not only language understanding, but also *knowledge* that supports sophisticated reasoning (Chen et al., 2016; Mi-haylov and Frank, 2018; Bauer et al., 2018; Zhong

et al., 2018). Thereby, we argue that pre-trained LMs, despite their powerfulness, could be further improved for MRC by integrating background knowledge. Fig. 1 gives a motivating example from ReCoRD (Zhang et al., 2018). In this example, the passage describes that Sudan faces trade sanctions from US due to its past support for North Korea. The cloze-style question states that Sudan is subject to the Trump’s ban, and asks the organization by which Sudan is deemed to be a state sponsor of terror. BERT fails on this case as there is not enough evidence in the text. But after introducing the world knowledge “*Trump is the person who leads US*” and word knowledge “*sanctions has a common hypernym with ban*”, we can reasonably infer that the answer is “*US*”. This example suggests the importance and necessity of integrating knowledge, even on the basis of a rather strong model like BERT. We refer interested readers to Appendix A for another motivating example from SQuAD1.1 (Rajpurkar et al., 2016).

Thus, in this paper, we devise KT-NET (abbr. for Knowledge and Text fusion NET), a new approach to MRC which improves pre-trained LMs with additional knowledge from knowledge bases (KBs). The aim here is to take full advantage of both linguistic regularities covered by deep LMs and high-quality knowledge derived from curated KBs, towards better MRC. We leverage two KBs: WordNet (Miller, 1995) that records lexical relations between words and NELL (Carlson et al., 2010) that stores beliefs about entities. Both are useful for the task (see Fig. 1). Instead of introducing symbolic facts, we resort to distributed representations (i.e., embeddings) of KBs (Yang and Mitchell, 2017). With such KB embeddings, we could (i) integrate knowledge relevant not only locally to the reading text but also globally about the whole KBs; and (ii) easily incorporate multiple KBs at the same time, with minimal task-specific engineering (see § 2.2 for detailed explanation).

As depicted in Fig. 2, given a question and passage, KT-NET first retrieves potentially relevant KB embeddings and encodes them in a knowledge memory. Then, it employs, in turn, (i) a BERT encoding layer to compute deep, context-aware representations for the reading text; (ii) a knowledge integration layer to select desired KB embeddings from the memory, and integrate them with BERT representations; (iii) a self-matching layer to fuse BERT and KB representations, so as to enable rich

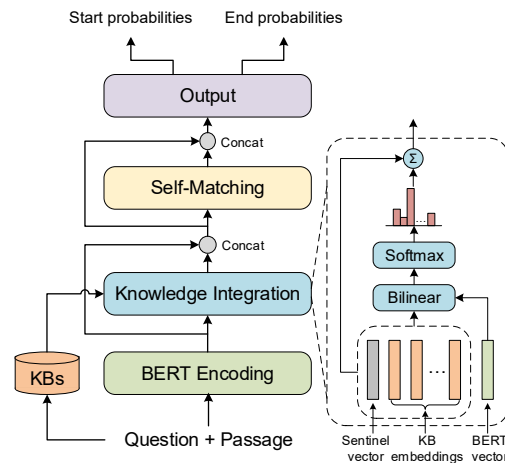


Figure 2: Overall architecture of KT-NET (left), with the knowledge integration module illustrated (right).

interactions among them; and (iv) an output layer to predict the final answer. In this way we enrich BERT with curated knowledge, combine merits of the both, and make knowledge-aware predictions.

We evaluate our approach on two benchmarks: ReCoRD (Zhang et al., 2018) and SQuAD1.1 (Rajpurkar et al., 2016). On ReCoRD, a passage is generated from the first few paragraphs of a news article, and the corresponding question the rest of the article, which, by design, requires background knowledge and reasoning. On SQuAD1.1 where the best models already outperform humans, questions remaining unsolved are really difficult ones. Both are appealing testbeds for evaluating genuine reading comprehension capabilities. We show that incorporating knowledge can bring significant and consistent improvements to BERT, which itself is one of the strongest models on both datasets.

The contributions of this paper are two-fold: (i) We investigate and demonstrate the feasibility of enhancing pre-trained LMs with rich knowledge for MRC. To our knowledge, this is the first study of its kind, indicating a potential direction for future research. (ii) We devise a new approach KT-NET to MRC. It outperforms competitive baselines, ranks the 1st place on the ReCoRD leaderboard, and is also the best single model on the SQuAD1.1 leaderboard at the time of submission (March 4th, 2019).

2 Our Approach

In this work we consider the *extractive* MRC task. Given a passage with m tokens $P = \{p_i\}_{i=1}^m$ and a question with n tokens $Q = \{q_j\}_{j=1}^n$, our goal


is to predict an answer A which is constrained as a contiguous span in the passage, i.e., $A = \{p_i\}_{i=a}^b$, with a and b indicating the answer boundary.

We propose KT-NET for this task, the key idea of which is to enhance BERT with curated knowledge from KBs, so as to combine the merits of the both. To encode knowledge, we adopt knowledge graph embedding techniques (Yang et al., 2015) and learn vector representations of KB concepts. Given passage P and question Q , we retrieve for each token $w \in P \cup Q$ a set of potentially relevant KB concepts $C(w)$, where each concept $c \in C(w)$ is associated with a learned vector embedding \mathbf{c} .

Based upon these pre-trained KB embeddings, KT-NET is built, as depicted in Fig. 2, with four major components: (i) a *BERT encoding layer* that computes deep, context-aware representations for questions and passages; (ii) a *knowledge integration layer* that employs an attention mechanism to select the most relevant KB embeddings, and integrates them with BERT representations; (iii) a *self-matching layer* that further enables rich interactions among BERT and KB representations; and (iv) an *output layer* that predicts the final answer. In what follows, we first introduce the four major components in § 2.1, and leave knowledge embedding and retrieval to § 2.2.

2.1 Major Components of KT-NET

KT-NET consists of four major modules: BERT encoding, knowledge integration, self-matching, and final output, detailed as follows.

BERT Encoding Layer  This layer uses BERT encoder to model passages and questions. It takes as input passage P and question Q , and computes for each token a context-aware representation.

Specifically, given passage $P = \{p_i\}_{i=1}^m$ and question $Q = \{q_j\}_{j=1}^n$, we first pack them into a single sequence of length $m + n + 3$, i.e.,

$$S = [\langle \text{CLS} \rangle, Q, \langle \text{SEP} \rangle, P, \langle \text{SEP} \rangle],$$

where $\langle \text{SEP} \rangle$ is the token separating Q and P , and $\langle \text{CLS} \rangle$ the token for classification (will not be used in this paper). For each token s_i in S , we construct its input representation as:

$$\mathbf{h}_i^0 = \mathbf{s}_i^{\text{tok}} + \mathbf{s}_i^{\text{pos}} + \mathbf{s}_i^{\text{seg}},$$

where $\mathbf{s}_i^{\text{tok}}$, $\mathbf{s}_i^{\text{pos}}$, and $\mathbf{s}_i^{\text{seg}}$ are the token, position, and segment embeddings for s_i , respectively. Tokens in Q share a same segment embedding \mathbf{q}^{seg} ,

and tokens in P a same segment embedding \mathbf{p}^{seg} . Such input representations are then fed into L successive Transformer encoder blocks, i.e.,

$$\mathbf{h}_i^\ell = \text{Transformer}(\mathbf{h}_i^{\ell-1}), \quad \ell = 1, 2, \dots, L,$$

so as to generate deep, context-aware representations for passages and questions. We refer readers to (Devlin et al., 2018; Vaswani et al., 2017) for details. The final hidden states $\{\mathbf{h}_i^L\}_{i=1}^{m+n+3} \in \mathbb{R}^{d_1}$ are taken as the output of this layer.

Knowledge Integration Layer This layer is designed to further integrate knowledge into BERT, and is a core module of our approach. It takes as input the BERT representations $\{\mathbf{h}_i^L\}$ output from the previous layer, and enriches them with relevant KB embeddings, which makes the representations not only context-aware but also knowledge-aware.

Specifically, for each token s_i , we get its BERT representation $\mathbf{h}_i^L \in \mathbb{R}^{d_1}$ and retrieve a set of potentially relevant KB concepts $C(s_i)$, where each concept c_j is associated with KB embedding $\mathbf{c}_j \in \mathbb{R}^{d_2}$. (We will describe the KB embedding and retrieval process later in § 2.2.) Then we employ an attention mechanism to adaptively select the most relevant KB concepts. We measure the relevance of concept c_j to token s_i with a bilinear operation, and calculate the attention weight as:

$$\alpha_{ij} \propto \exp(\mathbf{c}_j^\top \mathbf{W} \mathbf{h}_i^L), \quad (1)$$

where $\mathbf{W} \in \mathbb{R}^{d_2 \times d_1}$ is a trainable weight parameter. As these KB concepts are not necessarily relevant to the token, we follow (Yang and Mitchell, 2017) to further introduce a knowledge sentinel $\bar{\mathbf{c}} \in \mathbb{R}^{d_2}$, and calculate its attention weight as:

$$\beta_i \propto \exp(\bar{\mathbf{c}}^\top \mathbf{W} \mathbf{h}_i^L). \quad (2)$$

The retrieved KB embeddings $\{\mathbf{c}_j\}$ (as well as the sentinel $\bar{\mathbf{c}}$) are then aligned to s_i and aggregated accordingly, i.e.,

$$\mathbf{k}_i = \sum_j \alpha_{ij} \mathbf{c}_j + \beta_i \bar{\mathbf{c}}, \quad (3)$$

with $\sum_j \alpha_{ij} + \beta_i = 1$.² Here \mathbf{k}_i can be regarded as a knowledge state vector that encodes extra KB information w.r.t. the current token. We concatenate \mathbf{k}_i with the BERT representation \mathbf{h}_i^L and output $\mathbf{u}_i = [\mathbf{h}_i^L, \mathbf{k}_i] \in \mathbb{R}^{d_1+d_2}$, which is by nature not only context-aware but also knowledge-aware.

²We set $\mathbf{k}_i = \mathbf{0}$ if $C(s_i) = \emptyset$.

Self-Matching Layer This layer takes as input the knowledge-enriched representations $\{\mathbf{u}_i\}$, and employs a self-attention mechanism to further enable interactions among the context components $\{\mathbf{h}_i^L\}$ and knowledge components $\{\mathbf{k}_i\}$. It is also an important module of our approach.

We model both *direct* and *indirect* interactions. As for direct interactions, given two tokens s_i and s_j (along with their knowledge-enriched representations \mathbf{u}_i and \mathbf{u}_j), we measure their similarity with a trilinear function (Seo et al., 2017):

$$r_{ij} = \mathbf{w}^\top [\mathbf{u}_i, \mathbf{u}_j, \mathbf{u}_i \odot \mathbf{u}_j],$$

and accordingly obtain a similarity matrix \mathbf{R} with r_{ij} being the ij -th entry. Here \odot denotes element-wise multiplication, and $\mathbf{w} \in \mathbb{R}^{3d_1+3d_2}$ is a trainable weight parameter. Then, we apply a row-wise softmax operation on \mathbf{R} to get the self-attention weight matrix \mathbf{A} , and compute for each token s_i an attended vector \mathbf{v}_i , i.e.,

$$a_{ij} = \frac{\exp(r_{ij})}{\sum_j \exp(r_{ij})},$$

$$\mathbf{v}_i = \sum_j a_{ij} \mathbf{u}_j,$$

where a_{ij} is the ij -th entry of \mathbf{A} . \mathbf{v}_i reflects how each token s_j interacts directly with s_i .

Aside from direct interactions, indirect interactions, e.g., the interaction between s_i and s_j via an intermediate token s_k , are also useful. To further model such indirect interactions, we conduct a self-multiplication of the original attention matrix \mathbf{A} , and compute for each token s_i another attended vector $\bar{\mathbf{v}}_i$, i.e.,

$$\bar{\mathbf{A}} = \mathbf{A}^2,$$

$$\bar{\mathbf{v}}_i = \sum_j \bar{a}_{ij} \mathbf{u}_j,$$

where \bar{a}_{ij} is the ij -th entry of $\bar{\mathbf{A}}$. $\bar{\mathbf{v}}_i$ reflects how each token s_j interacts indirectly with s_i , through all possible intermediate tokens. Finally, we build the output for each token by a concatenation $\mathbf{o}_i = [\mathbf{u}_i, \mathbf{v}_i, \mathbf{u}_i - \mathbf{v}_i, \mathbf{u}_i \odot \mathbf{v}_i, \bar{\mathbf{v}}_i, \mathbf{u}_i - \bar{\mathbf{v}}_i] \in \mathbb{R}^{6d_1+6d_2}$.

Output Layer We follow BERT and simply use a linear output layer, followed by a standard softmax operation, to predict answer boundaries. The probability of each token s_i to be the start or end position of the answer span is calculated as:


$$p_i^1 = \frac{\exp(\mathbf{w}_1^\top \mathbf{o}_i)}{\sum_j \exp(\mathbf{w}_1^\top \mathbf{o}_j)}, \quad p_i^2 = \frac{\exp(\mathbf{w}_2^\top \mathbf{o}_i)}{\sum_j \exp(\mathbf{w}_2^\top \mathbf{o}_j)},$$


where $\{\mathbf{o}_i\}$ are output by the self-matching layer, and $\mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^{6d_1+6d_2}$ are trainable parameters. The training objective is the log-likelihood of the true start and end positions:

$$\mathcal{L} = -\frac{1}{N} \sum_{j=1}^N (\log p_{y_j^1}^1 + \log p_{y_j^2}^2),$$

where N is the number of examples in the dataset, and y_j^1, y_j^2 are the true start and end positions of the j -th example, respectively. At inference time, the span (a, b) where $a \leq b$ with maximum $p_a^1 p_b^2$ is chosen as the predicted answer.

2.2 Knowledge Embedding and Retrieval

Now we introduce the knowledge embedding and retrieval process. We use two KBs: WordNet and NELL, both stored as *(subject, relation, object)* triples, where each triple is a fact indicating a specific relation between two entities. WordNet stores lexical relations between word synsets, , (*organism, hypernym_of, animal*). NELL stores beliefs about entities, where the subjects are usually real-world entities and the objects are either entities, e.g., (*Coca Cola, headquartered_in, Atlanta*), or concepts, e.g., (*Coca Cola, is_a, company*). Below we shall sometimes abuse terminologies and refer to synsets, real-world entities, and concepts as “entities”. As we have seen in Fig. 1, both KBs are useful for MRC.

KB Embedding In contrast to directly encoding KBs as symbolic *(subject, relation, object)* facts, we choose to encode them in a continuous vector space. Specifically, given any triple (s, r, o) , we would like to learn vector embeddings of subject s , relation r , and object o , so that the validity of the triple can be measured in the vector space based on the embeddings. We adopt the BILINEAR model (Yang et al., 2015) which measures the validity  a bilinear function $f(s, r, o) = \mathbf{s}^\top \text{diag}(\mathbf{r}) \mathbf{o}$. Here, $\mathbf{s}, \mathbf{r}, \mathbf{o} \in \mathbb{R}^{d_2}$ are the vector embeddings associated with s, r, o , respectively, and $\text{diag}(\mathbf{r})$ is a diagonal matrix with the main diagonal given by \mathbf{r} . Triples already stored in a KB are supposed to have higher validity. A margin-based ranking loss is then accordingly designed to learn the embeddings (refer to (Yang et al., 2015) for details). After this embedding process, we obtain a vector representation for each entity (as well as relation) of the two KBs.

KB Concepts Retrieval In this work, we treat WordNet synsets and NELL concepts as knowl-

edge to be retrieved from KBs, similar to (Yang and Mitchell, 2017). For WordNet, given a passage or question word, we return its synsets as candidate KB concepts. For NELL, we first recognize named entities from a given passage and question, link the recognized mentions to NELL entities by string matching, and then collect the corresponding NELL concepts as candidates. Words within a same entity name and subwords within a same word will share the same retrieved concepts, e.g., we retrieve the NELL concept “*company*” for both “*Coca*” and “*Cola*”. After this retrieval process, we obtain a set of potentially relevant KB concepts for each token in the input sequence, where each KB concept is associated with a vector embedding.

Advantages Previous attempts that leverage extra knowledge for MRC (Bauer et al., 2018; Mihaylov and Frank, 2018) usually follow a retrieve-then-encode paradigm, i.e., they first retrieve relevant knowledge from KBs, and only the retrieved knowledge—which is relevant locally to the reading text—will be encoded and integrated for MRC. Our approach, by contrast, first learns embeddings for KB concepts with consideration of the whole KBs (or at least sufficiently large subsets of KBs). The learned embeddings are then retrieved and integrated for MRC, which are thus relevant not only locally to the reading text but also globally about the whole KBs. Such knowledge is more informative and potentially more useful for MRC.

Moreover, our approach offers a highly convenient way to simultaneously integrate knowledge from multiple KBs. For instance, suppose we retrieve for token s_i a set of candidate KB concepts $C^1(s_i)$ from WordNet, and $C^2(s_i)$ from NELL. Then, we can compute a knowledge state vector \mathbf{k}_i^1 based on $C^1(s_i)$, and \mathbf{k}_i^2 based on $C^2(s_i)$, which are further combined with the BERT hidden state \mathbf{h}_i^L to generate $\mathbf{u}_i = [\mathbf{h}_i^L, \mathbf{k}_i^1, \mathbf{k}_i^2]$. As such, \mathbf{u}_i naturally encodes knowledge from both KBs (see the knowledge integration layer for technical details).

3 Experiments

3.1 Datasets

In this paper we empirically evaluate our approach on two benchmarks: ReCoRD and SQuAD1.1.

ReCoRD—acronym for the Reading Comprehension with Commonsense Reasoning Dataset—is a large-scale MRC dataset requiring commonsense reasoning (Zhang et al., 2018). It consists

Dataset	Train	Dev	Test
ReCoRD	100,730	10,000	10,000
SQuAD1.1	87,599	10,570	9,533

Table 1: The number of training, development, and test examples of ReCoRD and SQuAD1.1.

of passage-question-answer tuples, collected from CNN and Daily Mail news articles. In each tuple, the passage is formed by the first few paragraphs of a news article, with named entities recognized and marked. The question is a sentence from the rest of the article, with a missing entity specified as the golden answer. The goal is to find the golden answer among the entities marked in the passage, which can be deemed as an extractive MRC task. This data collection process by design generates questions that require external knowledge and reasoning. It also filters out questions that can be answered simply by pattern matching, posing further challenges to current MRC systems. We take it as the major testbed for evaluating our approach.

SQuAD1.1 (Rajpurkar et al., 2016) is a well-known extractive MRC dataset that consists of questions created by crowdworkers for Wikipedia articles. The golden answer to each question is a span from the corresponding passage. In this paper, we focus more on answerable questions than unanswerable ones. Hence, we choose SQuAD1.1 rather than SQuAD2.0 (Rajpurkar et al., 2018).

Table 1 provides the statistics of ReCoRD and SQuAD1.1. On both datasets, the training and development (dev) sets are publicly available, but the test set is hidden. One has to submit the code to retrieve the final test score. As frequent submissions to probe the unseen test set are not encouraged, we only submit our best single model for testing,³ and conduct further analysis on the dev set. Both datasets use **Exact Match** (EM) and (macro-averaged) **F1** as the evaluation metrics (Zhang et al., 2018).

3.2 Experimental Setups

Data Preprocessing We first prepare pre-trained KB embeddings. We use the resources provided by Yang and Mitchell (2017), where the WordNet embeddings were pre-trained on a subset consisting of 151,442 triples with 40,943 synsets and 18 relations, and the NELL embeddings pre-trained on a subset containing 180,107 entities and 258

³In this paper, we restrict ourselves to improvements involving a single model, and hence do not consider ensembles.

concepts. Both groups of embeddings are 100-D. Refer to (Yang and Mitchell, 2017) for details.

Then we retrieve knowledge from the two KBs. For WordNet, we employ the BasicTokenizer built in BERT to tokenize text, and look up synsets for each word using NLTK (Bird and Loper, 2004). Synsets within the 40,943 subset are returned as candidate KB concepts for the word. For NELL, we link entity mentions to the whole KB, and return associated concepts within the 258 subset as candidate KB concepts. Entity mentions are given as answer candidates on ReCoRD, and recognized by Stanford CoreNLP (Manning et al., 2014) on SQuAD1.1.

Finally, we follow Devlin et al. (2018) and use the FullTokenizer built in BERT to segment words into wordpieces. The maximum question length is set to 64. Questions longer than that are truncated. The maximum input length ($|S|$) is set to 384. Input sequences longer than that are segmented into chunks with a stride of 128. The maximum answer length at inference time is set to 30.

Comparison Setting We evaluate our approach in three settings: KT-NET_{WordNet}, KT-NET_{NELL}, and KT-NET_{BOTH}, to incorporate knowledge from WordNet, NELL, and both of the two KBs, respectively. We take BERT as a direct baseline, in which only the BERT encoding layer and output layer are used, and no knowledge will be incorporated. Our BERT follows exactly the same design as the original paper (Devlin et al., 2018). Besides BERT, we further take top-ranked systems on each dataset as additional baselines (will be detailed in § 3.3).

Training Details For all three settings of KT-NET (as well as BERT), we initialize parameters of the BERT encoding layer with pre-trained models officially released by Google⁴. These models were pre-trained on the concatenation of BooksCorpus (800M words) and Wikipedia (2,500M words), using the tasks of masked language model and next sentence prediction (Devlin et al., 2018). We empirically find that the *cased, large* model—which is case sensitive and contains 24 Transformer encoding blocks, each with 16 self-attention heads and 1024 hidden units—performs the best on both datasets. Throughout our experiments, we use this setting unless specified otherwise. Other trainable parameters are randomly initialized.

⁴<https://github.com/google-research/bert>

Model	Dev		Test	
	EM	F1	EM	F1
Leaderboard (Mar. 4th, 2019)				
Human	91.28	91.64	91.31	91.69
#1 DCReader+BERT	—	—	70.49	71.98
#2 BERT _{BASE}	—	—	55.99	57.99
#3 DocQA w/ ELMo	44.13	45.39	45.44	46.65
#4 SAN	38.14	39.09	39.77	40.72
#5 DocQA	36.59	37.89	38.52	39.76
Ours				
BERT	70.22	72.16	—	—
KT-NET _{WordNet}	70.56	72.75	—	—
KT-NET _{NELL}	70.54	72.52	—	—
KT-NET _{BOTH}	71.60	73.61	73.01	74.76

Table 2: Results on ReCoRD. The top 5 systems are all single models and chosen for comparison.

Model	Dev		Test	
	EM	F1	EM	F1
Leaderboard (Mar. 4th, 2019)				
Human	80.3	90.5	82.30	91.22
#1 BERT+TriviaQA	84.2	91.1	85.08	91.83
#2 WD	—	—	84.40	90.56
#3 nlnet	—	—	83.47	90.13
#4 MARS	—	—	83.19	89.55
#5 QANet	—	—	82.47	89.31
Ours				
BERT	84.41	91.24	—	—
KT-NET _{WordNet}	85.15	91.70	85.94	92.43
KT-NET _{NELL}	85.02	91.69	—	—
KT-NET _{BOTH}	84.96	91.64	—	—

Table 3: Results on SQuAD1.1. The top 5 single models are chosen for comparison.

We use the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 3e-5 and a batch size of 24. The number of training epochs is chosen from {2,3,4}, according to the best EM+F1 score on the dev set of each dataset. During training, the pre-trained BERT parameters will be fine-tuned with other trainable parameters, and the KB embeddings will be kept fixed, which is empirically observed to offer the best performance.

3.3 Results

On ReCoRD and SQuAD1.1, we compare our approach to BERT and the top 5 (single) models on the leaderboard (exclusive of ours). The results are given in Table 2 and Table 3, respectively, where the scores of the non-BERT baselines are taken directly from the leaderboard and/or literature.

On ReCoRD⁵ (Table 2): (i) DCReader+BERT is the former top leaderboard system (unpublished) prior to our submission; (ii) BERT_{BASE} is BERT with the base setting (12 Transformer blocks, each

⁵<https://sheng-z.github.io/ReCoRD-explorer/>

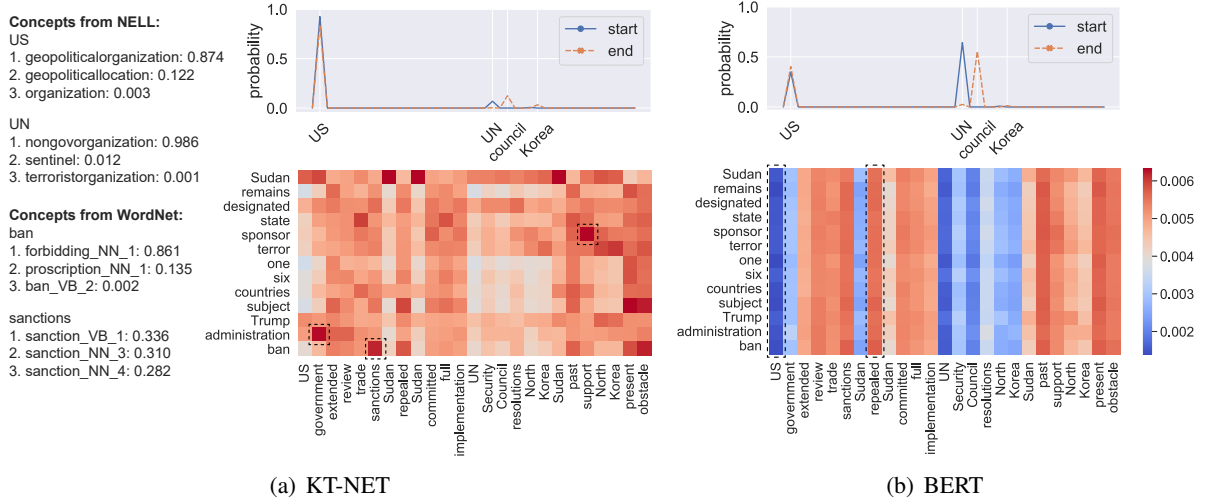


Figure 3: Case study. Heat maps present similarities between question (row) and passage (column) words. Line charts show probabilities of answer boundaries. In KT-NET, top 3 most relevant KB concepts are further given.

with 12 self-attention heads and 768 hidden units); (iii) DocQA (Liu et al., 2018) and SAN (Clark and Gardner, 2018) are two previous state-of-the-art MRC models; (iv) the pre-trained LM ELMo (Peters et al., 2018a) is further used in DocQA. All these models, except for DCReader+BERT, were re-implemented by the creators of the dataset and provided as official baselines (Zhang et al., 2018).

On SQuAD⁶ (Table 3): (i) BERT+TriviaQA is the former best model officially submitted by Google. It is an uncased, large model, and further uses data augmentation with TriviaQA (Joshi et al., 2017); (ii) WD, nlnet, and MARS are three competitive models that have not been published; (iii) QANet is a well performing MRC model proposed by Yu et al. (2018), and later re-implemented and submitted by Google Brain & CMU.

Results on dev sets show that (i) KT-NET consistently outperforms BERT (which itself already surpasses all the other baselines), irrespective of which KB is used, and on both datasets. Our best KT-NET model offers a 1.38/1.45 improvement in EM/F1 over BERT on ReCoRD, and a 0.74/0.46 improvement in EM/F1 on SQuAD1.1. (ii) Both KBs are capable of improving BERT for MRC, but the best setting varies across datasets. Integrating both KBs performs best on ReCoRD, while using WordNet alone is a better choice on SQuAD1.1.

Results on test sets further demonstrate the superiority of our approach. It significantly outperforms the former top leaderboard system by **+2.52 EM/+2.78 F1** on ReCoRD. And on SQuAD1.1,

although little room for improvement, it still gets a meaningful gain of **+0.86 EM/+0.60 F1** over the former best single model.

4 Case Study

This section provides a case study, using the motivating example described in Fig. 1, to vividly show the effectiveness of KT-NET, and make a direct comparison with BERT. For both methods, we use the optimal configurations that offer their respective best performance on ReCoRD (where the example comes from).

Relevant Knowledge Selection We first explore how KT-NET can adaptively select the most relevant knowledge w.r.t. the reading text. Recall that given a token s_i , the relevance of a retrieved KB concept c_j is measured by the attention weight α_{ij} (Eq. (1)), according to which we can pick the most relevant KB concepts for this token. Fig. 3(a) (left) presents 4 tokens from the question/passage, each associated with top 3 most relevant concepts from NELL or WordNet. As we can see, these attention distributions are quite meaningful, with “US” and “UN” attending mainly to the NELL concepts of “geopoliticalorganization” and “nongovorganization”, respectively, “ban” mainly to the WordNet synset “forbidding_NN_1”, and “sanction” almost uniformly to the three highly relevant synsets.

Question/Passage Representations We further examine how such knowledge will affect the final representations learned for the question/passage. We consider all sentences listed in Fig. 1, and con-

⁶<https://rajpurkar.github.io/SQuAD-explorer/>

tent words (nouns, verbs, adjectives, and adverbs) therein. For each word s_i , we take its final representation \mathbf{o}_i , obtained right before the output layer. Then we calculate the cosine similarity $\cos(\mathbf{o}_i, \mathbf{o}_j)$ between each question word s_i and passage word s_j . The resultant similarity matrices are visualized in Fig. 3(a) and Fig. 3(b) (heat maps), obtained by KT-NET and BERT, respectively.⁷

For BERT (Fig. 3(b)), given any passage word, all question words tend to have similar similarities to the given word, e.g., all the words in the question have a low degree of similarity to the passage word “US”, while a relatively high degree of similarity to “repealed”. Such phenomenon indicates that after fine-tuning in the MRC task, BERT tends to learn similar representations for question words, all of which approximately express the meaning of the whole question and are hard to distinguish.

For KT-NET (Fig. 3(a)), by contrast, different question words can exhibit diverse similarities to a passage word, and these similarities may perfectly reflect their relationships encoded in KBs. For example, we can observe relatively high similarities between: (i) “administration” and “government” which share a same synset, (ii) “ban” and “sanctions” which have a common hypernym, and (iii) “sponsor” and “support” where a synset of the former has the relation “*derivationally_related_form*” with the latter, all in WordNet. Such phenomenon indicates that after integrating knowledge, KT-NET can learn more accurate representations which enable better question-passage matching.

Final Answer Prediction Fig. 3(a) and Fig. 3(b) (line charts) list the probability of each word to be start/end of the answer, predicted by KT-NET and BERT, respectively. BERT mistakenly predicts the answer as “UN Security Council”, but our method successfully gets the correct answer “US”.

We observed similar phenomena on SQuAD1.1 and report the results in Appendix B.

5 Related Work

Machine Reading Comprehension In the last few years, a number of datasets have been created for MRC, e.g., CNN/DM (Hermann et al., 2015), SQuAD (Rajpurkar et al., 2016, 2018), SearchQA (Dunn et al., 2017), TriviaQA (Joshi et al., 2017), and MS-MARCO (Nguyen et al., 2016). These

datasets have led to advances like Match-LSTM (Wang and Jiang, 2017), BiDAF (Seo et al., 2017), AoA Reader (Cui et al., 2017), DCN (Xiong et al., 2017), R-Net (Wang et al., 2017), and QANet (Yu et al., 2018). These end-to-end neural models have similar architectures, starting off with an encoding layer to encode every question/passage word as a vector, passing through various attention-based interaction layers and finally a prediction layer.

More recently, LMs such as ELMo (Peters et al., 2018b), GPT (Radford et al., 2018), and BERT (Devlin et al., 2018) have been devised. They pre-train deep LMs on large-scale unlabeled corpora to obtain contextual representations of text. When used in downstream tasks including MRC, the pre-trained contextual representations greatly improve the performance in either a fine-tuning or feature-based way. Built upon pre-trained LMs, our work further explores the potential of incorporating structured knowledge from KBs, combining the strengths of both text and knowledge representations.

Incorporating KBs Several MRC datasets that require external knowledge have been proposed, such as ReCoRD (Zhang et al., 2018), ARC (Clark et al., 2018), MCScript (Ostermann et al., 2018), OpenBookQA (Mihaylov et al., 2018) and CommonsenseQA (Talmor et al., 2018). ReCoRD can be viewed as an extractive MRC dataset, while the later four are multi-choice MRC datasets, with relatively smaller size than ReCoRD. In this paper, we focus on the extractive MRC task. Hence, we choose ReCoRD and SQuAD in the experiments.

Some previous work attempts to leverage structured knowledge from KBs to deal with the tasks of MRC and QA. Weissenborn et al. (2017), Bauer et al. (2018), Mihaylov and Frank (2018), Pan et al. (2019), Chen et al. (2018), Wang et al. (2018) follow a retrieve-then-encode paradigm, i.e., they first retrieve relevant knowledge from KBs, and only the retrieved knowledge relevant locally to the reading text will be encoded and integrated. By contrast, we leverage pre-trained KB embeddings which encode whole KBs. Then we use attention mechanisms to select and integrate knowledge that is relevant locally to the reading text. Zhong et al. (2018) try to leverage pre-trained KB embeddings to solve the multi-choice MRC task. However, the knowledge and text modules are not integrated, but used independently to predict the answer. And the model cannot be applied to extractive MRC.

⁷During visualization, we use a row-wise softmax operation to normalize similarity scores over all passage tokens.

6 Conclusion

This paper introduces KT-NET for MRC, which enhances BERT with structured knowledge from KBs and combines the merits of the both. We use two KBs: WordNet and NELL. We learn embeddings for the two KBs, select desired embeddings from them, and fuse the selected embeddings with BERT hidden states, so as to enable context- and knowledge-aware predictions. Our model achieves significant improvements over previous methods, becoming the best single model on ReCoRD and SQuAD1.1 benchmarks. This work demonstrates the feasibility of further enhancing advanced LMs with knowledge from KBs, which indicates a potential direction for future research.

Acknowledgments

This work is supported by the Natural Science Foundation of China (No.61533018, 61876009 and 61876223) and Baidu-Peking University Joint Project. We would like to thank the ReCoRD and SQuAD teams for evaluating our results on the anonymous test sets. And we are grateful to Tom M. Mitchell and Bishan Yang for sharing with us the valuable KB resources. We would also like to thank the anonymous reviewers for their insightful suggestions.

References

- Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4220–4230. Association for Computational Linguistics.
- Steven Bird and Edward Loper. 2004. Nltk: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, page 31. Association for Computational Linguistics.
- Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pages 1306–1313. AAAI Press.
- Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the cnn/daily mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367. Association for Computational Linguistics.
- Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. 2018. Neural natural language inference models enhanced with external knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2406–2417. Association for Computational Linguistics.
- Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855. Association for Computational Linguistics.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the AI2 reasoning challenge. *arXiv e-prints*, arXiv:1803.05457.
- Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. 2017. Attention-over-attention neural networks for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 593–602. Association for Computational Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv e-prints*, arXiv:1810.04805.
- Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv e-prints*, arXiv:1704.05179.
- Yoav Goldberg. 2019. Assessing bert’s syntactic abilities. *arXiv e-prints*, arXiv:1901.05287.
- Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28*, pages 1693–1701. Curran Associates, Inc.
- Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611. Association for Computational Linguistics.
- Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.
- Xiaodong Liu, Yelong Shen, Kevin Duh, and Jianfeng Gao. 2018. Stochastic answer networks for machine reading comprehension. In *Proceedings of the*

- 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1694–1704. Association for Computational Linguistics.
- Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60. Association for Computational Linguistics.
- Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391. Association for Computational Linguistics.
- Todor Mihaylov and Anette Frank. 2018. Knowledgeable reader: Enhancing cloze-style reading comprehension with external commonsense knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 821–832. Association for Computational Linguistics.
- George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.
- Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating Neural and Symbolic Approaches*, pages 96–105.
- Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. 2018. Semeval-2018 task 11: Machine comprehension using commonsense knowledge. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 747–757. Association for Computational Linguistics.
- Xiaoman Pan, Kai Sun, Dian Yu, Heng Ji, and Dong Yu. 2019. Improving question answering with external knowledge. *arXiv e-prints*, arXiv:1902.00993.
- Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018a. Deep contextualized word representations. 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 2227–2237. Association for Computational Linguistics.
- Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018b. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509. Association for Computational Linguistics.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Technical report, Technical report, OpenAI.
- Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don’t know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789. Association for Computational Linguistics.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392. Association for Computational Linguistics.
- Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *International Conference on Learning Representations (ICLR)*.
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv e-prints*, arXiv:1811.00937.
- 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. Curran Associates, Inc.
- Liang Wang, Meng Sun, Wei Zhao, Kewei Shen, and Jingming Liu. 2018. Yuanfudao at semeval-2018 task 11: Three-way attention and relational knowledge for commonsense machine comprehension. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 758–762. Association for Computational Linguistics.
- Shuohang Wang and Jing Jiang. 2017. Machine comprehension using match-lstm and answer pointer. In *International Conference on Learning Representations (ICLR)*.
- Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 189–198. Association for Computational Linguistics.
- Dirk Weissenborn, Tomáš Kočiský, and Chris Dyer. 2017. Dynamic integration of background knowledge in neural NLU systems. *arXiv e-prints*, arXiv:1706.02596.

Caiming Xiong, Victor Zhong, and Richard Socher. 2017. Dynamic coattention networks for question answering. In *International Conference on Learning Representations (ICLR)*.

Bishan Yang and Tom Mitchell. 2017. Leveraging knowledge bases in lstms for improving machine reading. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1436–1446. Association for Computational Linguistics.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *International Conference on Learning Representations (ICLR)*.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. In *International Conference on Learning Representations (ICLR)*.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv e-prints*, arXiv:1810.12885.

WanJun Zhong, Duyu Tang, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2018. Improving question answering by commonsense-based pre-training. *arXiv e-prints*, arXiv:1809.03568.

A Motivating Example from SQuAD1.1

We provide a motivating example from SQuAD1.1 to show the importance and necessity of integrating background knowledge. We restrict ourselves to knowledge from WordNet, which offers the best performance on this dataset according to our experimental results (Table 3).

Fig. 4 presents the example. The passage states that the congress aimed to formalize a unified front in trade and negotiations with various Indians, but the plan was never ratified by the colonial legislatures nor approved of by the crown. And the question asks whether the plan was formalized. BERT fails on this case by spuriously matching the two “formalize” appearing in the passage and question. But after introducing the word knowledge “*ratified is a hypernym of formalized*” and “*approved has a common hypernym with formalized*”, we can successfully predict that the correct answer is “*never ratified by the colonial legislatures nor approved of by the crown*”.

Passage: [...] The goal of the congress was to formalize a unified front in trade and negotiations with various Indians, since allegiance of the various tribes and nations was seen to be pivotal in the success in the war that was unfolding. The plan that the delegates agreed to was never **ratified** by the colonial legislatures nor **approved** of by the crown. [...]

Question: Was the plan **formalized**?

Original BERT prediction: formalize a unified front in trade and negotiations with various Indians

Prediction with background knowledge: never ratified by the colonial legislatures nor approved of by the crown

Background knowledge:

(ratified, hypernym-of, formalized)

(approved, common-hypernym-with, formalized)

Figure 4: An example from SQuAD1.1. The vanilla BERT model fails to predict the correct answer. But it succeeds after integrating background knowledge collected from WordNet.

B Case Study on SQuAD1.1

We further provide a case study, using the above example, to vividly show the effectiveness of our method KT-NET, and make a direct comparison with BERT. We use the same analytical strategy as described in § 4. For both KT-NET and BERT, we use the optimal configurations that offer their respective best performance on SQuAD1.1 (where the example comes from).

Relevant Knowledge Selection We first explore how KT-NET can adaptively select the most relevant knowledge w.r.t. the reading text. Fig.5(a) (left) presents 3 words from the question/passage, each associated with top 3 most relevant synsets from WordNet.⁸ Here the relevance of synset c_j to word s_i is measured by the attention weight α_{ij} (Eq. (1)).⁹ As we can see, these attention distributions are quite meaningful, with “*ratified*” attending mainly to WordNet synset “*sign_VB_2*”, “*formalized*” mainly to synset “*formalize_VB_1*”, and “*approved*” mainly to synsets “*approve_VB_2*” and “*sanction_VB_1*”.

Question/Passage Representations We further examine how such knowledge will affect the final representations learned for the question/passage. We consider all sentences listed in Fig. 4, and content words (nouns, verbs, adjectives, and adverbs) therein. For each word s_i , we take its final repre-

⁸We retrieve a single synset “*sign_VB_2*” for “*ratified*”.

⁹If word s_i consists of multiple subwords, we average the relevance of c_j over these subwords.

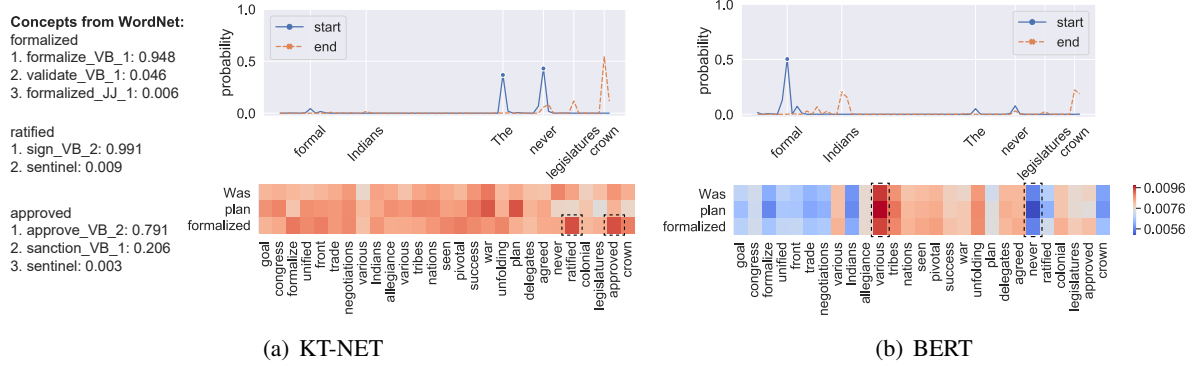


Figure 5: Case study. Heat maps present similarities between question (row) and passage (column) words. Line charts show probabilities of answer boundaries. In KT-NET, top 3 most relevant KB concepts are further given.

sensation \mathbf{o}_i , obtained right before the output layer. Then we calculate the cosine similarity $\cos(\mathbf{o}_i, \mathbf{o}_j)$ between each question word s_i and passage word s_j . The resultant similarity matrices are visualized in Fig.5(a) and Fig.5(b) (heat maps), obtained by KT-NET and BERT, respectively.¹⁰

For BERT (Fig.5(b)), we observe very similar patterns as in the ReCoRD example (§ 4). Given any passage word, all question words tend to have similar similarities to the given word, e.g., all the words in the question have a low degree of similarity to the passage word “*never*”, while a relatively high degree of similarity to “*various*”. Such phenomenon indicates, again, that after fine-tuning in the MRC task, BERT tends to learn similar representations for question words, all of which approximately express the meaning of the whole question and are hard to distinguish.

For KT-NET (Fig.5(a)), although the similarities between question and passage words are generally higher, these similarities may still perfectly reflect their relationships encoded in KBs. For example, we can observe relatively high similarities between: (i) “*formalized*” and “*ratified*” where the latter is a hypernym of the former; (ii) “*formalized*” and “*approved*” which share a common hypernym in WordNet. Such phenomenon indicates, again, that after integrating knowledge, KT-NET can learn more accurate representations which enable better question-passage matching.

Final Answer Prediction With the learned representations, predicting final answers is a natural next step. Fig.5(a) and Fig.5(b) (line charts) list

the probability of each word to be the start/end of the answer, predicted by KT-NET and BERT, respectively. As we can see, BERT mistakenly predicts the answer as “*formalize a unified front in trade and negotiations with various Indians*”, but our method successfully gets the correct answer “*never ratified by the colonial legislatures nor approved of by the crown*”.

The phenomena observed here are quite similar to those observed in the ReCoRD example, both demonstrating the effectiveness of our method and its superiority over BERT.

¹⁰During visualization, we take the averaged cosine similarity if word s_i or word s_j has subwords. And we use a row-wise softmax operation to normalize similarity scores over all passage tokens.