KEML: A Knowledge-Enriched Meta-Learning Framework for Lexical Relation Classification

Chengyu Wang,[1] **Minghui Qiu,**[1*] **Jun Huang,**[1] **Xiaofeng He**[2]

[1] Alibaba Group

[2] School of Computer Science and Technology, East China Normal University

{chengyu.wcy, minghui.qmh, huangjun.hj}@alibaba-inc.com, hexf@cs.ecnu.edu.cn

Abstract

Lexical relations describe how concepts are semantically related, in the form of relation triples. The accurate prediction of lexical relations between concepts is challenging, due to the sparsity of patterns indicating the existence of such relations. We propose the Knowledge-Enriched Meta-Learning (KEML) framework to address lexical relation classification. In KEML, the LKB-BERT (Lexical Knowledge Base-BERT) model is first presented to learn concept representations from text corpora, with rich lexical knowledge injected by distant supervision. A probabilistic distribution of auxiliary tasks is defined to increase the model's ability to recognize different types of lexical relations. We further propose a neural classifier integrated with special relation recognition cells, in order to combine meta-learning over the auxiliary task distribution and supervised learning for LRC. Experiments over multiple datasets show KEML outperforms state-of-the-art methods.

Introduction

As an important type of linguistic resources, lexical relations describe semantic associations between concepts, which are organized as backbones in lexicons, semantic networks and knowledge bases. The explicit usage of such resources has benefited a variety of NLP tasks, including relation extraction (Shen et al. 2018), question answering (Yang et al. 2017) and machine translation (Thompson et al. 2019).

To accumulate such knowledge, Lexical Relation Classification (LRC) is a basic NLP task to classify concepts into a finite set of lexical relations. Pattern-based and distributional methods are two major types of LRC models (Shwartz and Dagan 2016; Wang, He, and Zhou 2017). Compared to the classification of *factual relations* for knowledge graph population (Liu et al. 2017; Wu and He 2019), the accurate classification of lexical relations is more challenging. i) Most lexical relations represent the *common sense of human knowledge*, not frequently expressed in texts explicitly[1]. Apart from Hearst patterns (Hearst 1992) for hypernymy ("*is-a*") extraction, textual patterns that indicate the existence of other types of lexical relations remain few, leading to the "*pattern sparsity*" problem (Shwartz and Dagan 2016; Washio and Kato 2018a). ii) Distributional models assume concepts with similar contexts have similar embeddings (Mikolov et al. 2013). Representations of a concept pair learned by traditional word embedding models are not sufficient to distinguish different types of lexical relations (Glavas and Vulic 2018; Ponti et al. 2019). iii) Many LRC datasets are *highly imbalanced* w.r.t. training instances of different lexical relations, and may contain *randomly paired* concepts. It is difficult for models to distinguish whether a concept pair has a particular type of lexical relation, or has *very weak or no semantic relatedness*.

In this paper, we present the *Knowledge-Enriched Meta-Learning* (KEML) framework to address these challenges of LRC. KEML consists of three modules: *Knowledge Encoder*, *Auxiliary Task Generator* and *Relation Learner*. In *Knowledge Encoder*, we propose the LKB-BERT (Lexical Knowledge Base-BERT) model to learn *relation-sensitive concept representations*. LKB-BERT is built upon BERT (Devlin et al. 2019) and trained via *new distantly supervised learning tasks* over lexical knowledge bases, encoding both contextual representations and relational lexical knowledge. In *Auxiliary Task Generator*, we treat recognizing single type of lexical relations as *auxiliary tasks*. Based on meta-learning (Finn, Abbeel, and Levine 2017), a probabilistic distribution of auxiliary tasks is properly defined for the model to optimize, which addresses the imbalanced property and the existence of random relations in LRC datasets. In *Relation Learner*, we design a feed-forward neural network for LRC, of which the training process combines both gradient-based meta-learning and supervised learning. Especially, a special relation recognition cell is designed and integrated into the network for the purpose.

This paper makes the following contributions:

- We present LKB-BERT to learn relation-sensitive contextual representations of concepts by leveraging rich relational knowledge in knowledge bases.
- A meta-learning process combined with an auxiliary task distribution is defined to improve the ability of distributional LRC models to recognize lexical relations.
- We design a feed-forward neural network combined with special relation recognition cells to accommodate both

*Corresponding author.

[1] For example, "(car, meronymy, steering wheel)" can be paraphrased as "steering wheels are part of cars". However, the expression is usually omitted in texts, since it is common sense to humans.

meta-learning and supervised learning for LRC.

- In the experiments, we evaluate the KEML model over multiple public benchmark datasets. Experimental results show that KEML outperforms state-of-the-art methods.

Related Work

In this section, we overview the related work on LRC, pre-trained language models and meta-learning.

Lexical Relation Classification. As summarized in Shwartz and Dagan (2016), LRC models are categorized into two major types: *pattern-based* and *distributional*. Pattern-based approaches extract patterns w.r.t. a concept pair from texts as features to predict its lexical relation. For hypernymy relations, Hearst patterns (Hearst 1992) are most influential, often used for the construction of large-scale taxonomies (Wu et al. 2012). To learn patterns representations, Shwartz, Goldberg, and Dagan (2016) exploit LSTM-based RNNs to encode dependency paths of patterns. Roller, Kiela, and Nickel (2018); Le et al. (2019) calculate Hearst pattern-based statistics from texts to predict the degrees of hypernymy between concepts. For other types of lexical relations, LexNET (Shwartz and Dagan 2016) extends the network architecture (Shwartz, Goldberg, and Dagan 2016) for multi-way classification of lexical relations. Nguyen, Schulte im Walde, and Vu (2016, 2017) design path-based neural networks to distinguish antonymy and synonymy. However, these methods may suffer from the lack of patterns and concept pair occurrence in texts (Washio and Kato 2018a).

With the rapid development of deep learning, distributional models attract more interest. While traditional methods directly leverage the two concepts' embeddings as features for the classifier (Weeds et al. 2014; Vylomova et al. 2016; Roller and Erk 2016), they may suffer from the "*lexical memorization*" problem (Levy et al. 2015). Recently, more complicated neural networks are proposed. Attia et al. (2016) formulate LRC as a multi-task learning task and propose a convolutional neural network for LRC. Mrksic et al. (2017) propose the Attract-Repe model to learn the semantic specialization of word embeddings. Glavas and Vulic (2018) introduce the Specialization Tensor Model, which learns multiple relation-sensitive specializations of concept embeddings. SphereRE (Wang, He, and Zhou 2019) encodes concept pairs in the hyperspherical embedding space and achieves state-of-the-art results. A few works learn word-pair representations for other NLP tasks (Washio and Kato 2018b; Joshi et al. 2019; Camacho-Collados, Anke, and Schockaert 2019). KEML is also distributional, improving LRC by training meta-learners over language models.

Pre-trained Language Models. Pre-trained language models have gained attention from the NLP community (Qiu et al. 2020). ELMo (Peters et al. 2018) learns context-sensitive embeddings for each token form both left-to-right and right-to-left. BERT (Devlin et al. 2019) employs layers of transformer encoders to learn language representations. Follow-up works include Transformer-XL (Dai et al. 2019), XLNet (Yang et al. 2019), ALBERT (Lan et al. 2019) and many more. Yet another direction is to fuse additional knowledge sources into BERT-like models. ERNIE (Zhang et al. 2019) incorporates the rich semantics of entities in the model. KG-BERT (Yao, Mao, and Luo 2019) and K-BERT (Liu et al. 2019) employ relation prediction objectives in knowledge graphs as additional learning tasks. Xiong et al. (2020) propose to pre-train neural language models with online encyclopedias. In our work, we leverage the conceptual facts in lexical knowledge bases to improve the representation learning for LRC.

Meta-learning. Meta-learning is a learning paradigm to train models that can adapt to a variety of different tasks with little training data (Vanschoren 2018), mostly applied to few-shot learning (typically by N-way K-shot). In NLP, meta-learning algorithms have not been extensively employed, mostly due to the large numbers of training examples required to train the model for different NLP tasks. Existing models mostly focus on training meta-learners for single applications, such as link prediction (Chen et al. 2019), dialog systems (Madotto et al. 2019) and semantic parsing (Guo et al. 2019). Dou, Yu, and Anastasopoulos (2019) leverage meta-learning for various low-resource natural language understanding tasks.

To our knowledge, KEML is one of the early attempts to improve LRC by meta-learning (Finn, Abbeel, and Levine 2017). Different from the traditional N-way K-shot setting, our work employs the meta-learning framework as an intermediate step to enhance the capacity of the relation classifier. Since the mechanism of meta-learning is not our major research focus, we do not further elaborate.

The KEML Framework

In this section, we formally describe the KEML framework. A brief technical flow is presented, followed by its details.

A Brief Overview of KEML

We first overview the LRC task briefly. Denote (x_i, y_i) as an arbitrary concept pair. The goal of LRC is to learn a classifier f to predict the lexical relation $r_i \in \mathcal{R}$ between x_i and y_i, based on a labeled, training set $\mathcal{D} = \{(x_i, y_i, r_i)\}$. Here, $\mathcal{R}$ is the collection of all pre-defined lexical relation types (e.g., hypernymy, synonymy), (possibly) including a special relation type $\mathbf{RAN}$ ("*random*"), depending on different task settings. It means that x_i and y_i are randomly paired, without clear association with any lexical relations.

The framework of KEML is illustrated in Figure 1, with three modules introduced below:

Knowledge Encoder. Representation learning for LRC is significantly different from learning traditional word embeddings. This is because some lexical concepts are naturally *Multiword Expressions* (e.g., *card game*, *orange juice*), in which the entire sequences of tokens should be encoded in the embedding space (Cordeiro et al. 2016). Additionally, these models are insufficient to capture the lexical relations between concepts, due to the pattern sparsity issue (Washio and Kato 2018a). Hence, the semantics of concepts should be encoded from a larger corpus and rich language resources. Inspired by BERT (Devlin et al. 2019), and its extensions, we propose the LKB-BERT (Lexical Knowledge Base-BERT) model to encode the semantics of concepts

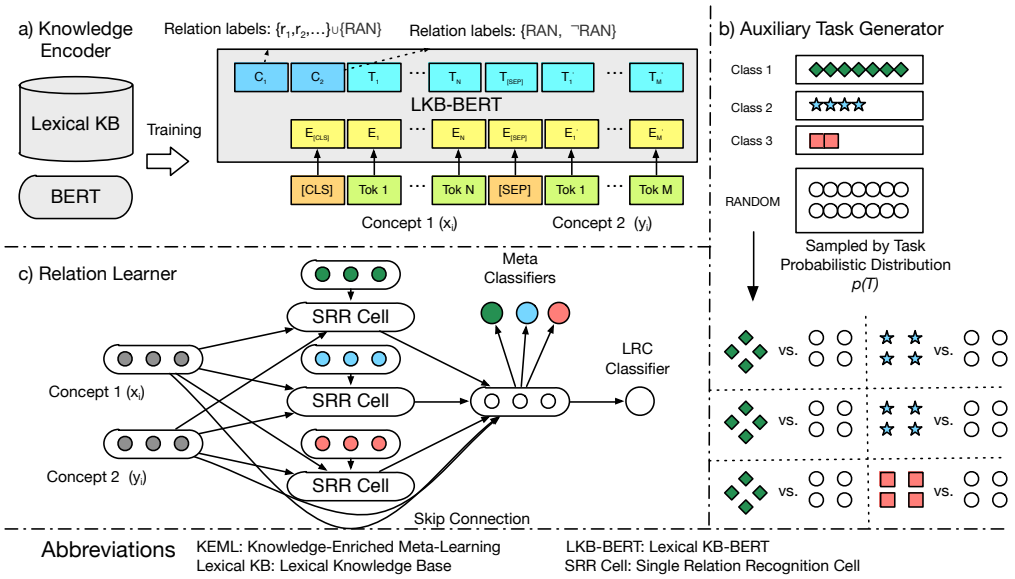


Figure 1: The high-level framework of KEML (best viewed in color).

from massive text corpora and lexical knowledge bases. After the training of LKB-BERT, each concept x_i or y_i receives the embeddings of the last transformer encoding layer of LKB-BERT as its representation. Denote the embeddings of x_i (or y_i) as $\vec{x}_i$ (or $\vec{y}_i$), with the dimension as d.

Auxiliary Task Generator. As discussed in related work, training embedding based classifiers directly may produce sub-optimal results due to i) the *lexical memorization* problem (Levy et al. 2015) and ii) the *highly imbalanced* nature of LRC training sets and the existence of **RAN** relations. Inspired by the design philosophy of meta-learning (Finn, Abbeel, and Levine 2017; Finn, Xu, and Levine 2018) and its NLP applications (Chen et al. 2019; Madotto et al. 2019), before we learn the LRC model, we design a series of auxiliary tasks for the model to solve. Each task aims at distinguishing between concept pairs that have a particular relation $r \in \mathcal{R}$ and randomly paired concepts. By solving these tasks, the network can gradually learn how to identify each type of lexical relation r before the final relation classifier is trained. The training procedure is discussed next.

Relation Learner. We design a two-stage algorithm to train a neural network based relation classifier f: i) meta-learning and ii) supervised learning. In the meta-learning stage, the adapted model parameters are iteratively learned by solving auxiliary tasks. Hence, the neural network learns how to recognize specific lexical relations, with the guidance of the lexical knowledge base. In the supervised learning stage, we fine-tune meta-learned parameters to obtain the multi-way classification model for LRC over $\mathcal{D}$.

Knowledge Encoder

The LKB-BERT model has the similar network architecture to that of BERT (Devlin et al. 2019). In original BERT, the inputs are arbitrary spans of token sequences. To encode the semantics of concept pairs, we combine a concept pair (x_i, y_i) to form a sequence of tokens, separated by a special token "[SEP]" as the input for LKB-BERT (see Figure 1). We first initialize all the model parameters of transformer encoders to be the same as BERT's pre-training results (apart from the output layer). Different from any standard fine-tuning tasks in BERT (Devlin et al. 2019), to address the **RAN** problem, LKB-BERT has two output classifiers. The first one classifies a concept pair (x_i, y_i) into the lexical relation collection $\mathcal{R}$ (including the special relation type **RAN**). Let $\mathcal{KB}$ be the collection of labeled concept pairs in lexical knowledge bases. For each concept pair with its label $(x_i, y_i, r_i) \in \mathcal{KB}$, we compute $\tau_r(x_i, y_i)$ as the predicted score w.r.t. the lexical relation r by LKB-BERT's transformer encoders (we have $\forall r \in \mathcal{R}, \tau_r(x_i, y_i) \in [0, 1]$ and $\sum_{r\in\mathcal{R}} \tau_r(x_i, y_i) = 1$). The first loss, i.e., the multi-way relation classification loss $\mathcal{L}_{KB}^{(1)}$ is defined as follows:[2]

$$\mathcal{L}_{KB}^{(1)} = -\sum_{(x_i,y_i,r_i)\in\mathcal{KB}} \sum_{r\in\mathcal{R}} (\mathbf{1}(r_i = r) \cdot \log \tau_r(x_i, y_i))$$

where $\mathbf{1}(\cdot)$ is the indicator function that returns 1 if the input expression is true; and 0 otherwise.

To improve LKB-BERT's ability to recognize concept pairs without any lexical relations, we add a binary cross-entropy loss for LKB-BERT to optimize. We only need LKB-BERT to learn whether a concept pair (x_i, y_i) are randomly paired. Let $\neg$**RAN** be any non-random lexical relation type in $\mathcal{R}$. The complete objective of LKB-BERT is: $\mathcal{L}_{KB} = \mathcal{L}_{KB}^{(1)} + \mathcal{L}_{KB}^{(2)}$, with $\mathcal{L}_{KB}^{(2)}$ to be:

$$\mathcal{L}_{KB}^{(2)} = -\sum_{(x_i,y_i,r_i)\in\mathcal{KB}} (\mathbf{1}(r_i = \mathbf{RAN}) \cdot \log \tau_{\mathbf{RAN}}(x_i, y_i) + \mathbf{1}(r_i = \neg\mathbf{RAN}) \cdot \log \tau_{\neg\mathbf{RAN}}(x_i, y_i))$$

[2]For simplicity, we omit all the regularization terms of objective functions throughout this paper.

In KEML, we regard lexical relations sampled from WordNet (Miller 1995) and the training set that we use to train the final relation classifier as the sources of $\mathcal{KB}$. Readers can also refer to the C_1 and C_2 units of LKB-BERT in Figure 1.

Auxiliary Task Generator

Although LKB-BERT is capable of learning deep concept representations, using such features for classifier training is insufficient. The reasons are twofold. i) Direct classification can suffer from "lexical memorization" (Levy et al. 2015), meaning that the relation classifier f only learns the individual characteristics of each one of the two concepts alone. ii) The LRC datasets are highly imbalanced. For example, in the widely used dataset EVALution (Santus et al. 2015), the numbers of training instances w.r.t. several lexical relation types are very few. Hence, the learning bias of the classifier trained by naive approaches is almost unavoidable.

The small number of training instances of certain lexical relations motivates us to employ few-shot meta-learning to address this issue (Finn, Abbeel, and Levine 2017), instead of multi-task learning of all the lexical relations. In KEML, we propose a series of auxiliary tasks $\mathcal{T}$, where each task (named *Single Relation Recognition*) $\mathcal{T}_r \in \mathcal{T}$ corresponds to a specific type of lexical relation $r \in \mathcal{R}$ (excluding $\mathbf{RAN}$). The goal is to distinguish concept pairs with the lexical relation type r and randomly paired concepts. Let $\mathcal{S}_r$ and $\mathcal{S}_{\mathbf{RAN}}$ be the collection of concept pairs with lexical relations as r and $\mathbf{RAN}$, respectively, randomly sampled from the training set $\mathcal{D}$. The goal of learning auxiliary task $\mathcal{T}_r$ is to minimize the following loss function $\mathcal{L}(\mathcal{T}_r)$:

$$\mathcal{L}(\mathcal{T}_r) = -\sum_{(x_i,y_i,r_i)\in\mathcal{S}_r\cup\mathcal{S}_{\mathbf{RAN}}} (\mathbf{1}(r_i = r) \cdot \log q_r(x_i, y_i) + \mathbf{1}(r_i = \mathbf{RAN}) \cdot \log q_{\mathbf{RAN}}(x_i, y_i))$$

where $q_r(x_i, y_i)$ is the predicted probability of the concept pair (x_i, y_i) having the lexical relation r. By solving the tasks $\mathcal{T}$, the model gradually learns how to recognize all these lexical relations, and to identify whether there exists any lexical relation between the two concepts at all.

A remaining problem is the design of the probabilistic distribution of auxiliary tasks $p(\mathcal{T})$. We need to consider two issues. i) The semantics of all types of lexical relations should be fully learned. ii) Assume the batch sizes for all tasks are the same, i.e., $\forall r_p, r_q \in \mathcal{R} \setminus \{\mathbf{RAN}\}, |\mathcal{S}_{r_p}| = |\mathcal{S}_{r_q}|$. Tasks related to lexical relations with more training instances should be learned more frequently by the meta-learner. Let $\mathcal{D}_r$ be the subset of the training set $\mathcal{D}$ with the lexical relation as r. $\forall r_p, r_q \in \mathcal{R} \setminus \{\mathbf{RAN}\}$, if $|\mathcal{D}_{r_p}| > |\mathcal{D}_{r_q}|$, we have the sampling probability $p(\mathcal{T}_{r_p}) > p(\mathcal{T}_{r_q})$. Hence, we define $p(\mathcal{T}_{r_p})$ empirically as follows:

$$p(\mathcal{T}_r) = \frac{\ln|\mathcal{D}_r| + \gamma}{\sum_{r'\in\mathcal{R}\setminus\{\mathbf{RAN}\}}(\ln|\mathcal{D}_{r'}| + \gamma)}$$

where $\gamma > 0$ is the smoothing factor. This setting oversamples lexical relations with few training instances, alleviating the imbalanced class issue. Overall, the expectation of all the losses of auxiliary tasks (represented as $\mathcal{L}(\mathcal{T})$) is: $\mathbb{E}(\mathcal{L}(\mathcal{T})) = \sum_{r^*\in\mathcal{R}\setminus\{\mathbf{RAN}\}} p(\mathcal{T}_r) \cdot \mathcal{L}(\mathcal{T}_r)$, which is the real learning objective that these auxiliary tasks aim to optimize.

Relation Learner

We now introduce the learning algorithm and the network structure for LRC. Assume the classifier f is parameterized by θ, with learning and meta-learning rates as α and ϵ. Relation Learner has two stages: i) meta-learning and ii) supervised learning. For each iteration in meta-learning, we sample N auxiliary tasks from $p(\mathcal{T})$. For each auxiliary task $\mathcal{T}_r$, we learn adapted parameters based on two sampled subsets: $\mathcal{S}_r$ and $\mathcal{S}_{\mathbf{RAN}}$, to make the model to recognize one specific relation type. After that, the adapted parameters on each task $\mathcal{T}_r$ are averaged and updated to θ. We simplify the meta-update step by only taking first-order derivatives (Nichol, Achiam, and Schulman 2018) to avoid the time-consuming second-order derivative computation. For supervised learning, we fine-tune the parameters θ of the classifier f to obtain the multi-way LRC model over the entire training set $\mathcal{D}$. The algorithmic description is shown in Algorithm 1.

Algorithm 1 Meta-Learning Algorithm for LRC

```
Initialize model parameters θ;
while not converge do
    Sample N auxiliary tasks T_{r_1}, T_{r_2}, ..., T_{r_N} from the task distribution p(T);
    for each auxiliary task T_r do
        Sample a batch (positive samples S_r and negative samples S_RAN) from the training set D;
        Update adapted parameters: θ_r ← θ − α∇L(T_r) based on S_r and S_RAN;
    end for
    Update meta-parameters: θ ← θ − ε∇ Σ_{T_r} L(T_r);
end while
Fine-tune θ over D by standard supervised learning LRC;
```

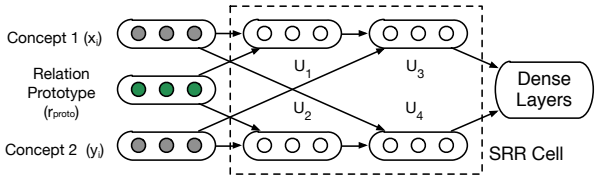


Figure 2: Structure of SRR Cell (we only show one cell, with some other parts of the network omitted).

Finally, we describe the neural network structure for LRC. In this network, the *Single Relation Recognition Cell* (SRR Cell) is designed for learning auxiliary tasks and enabling knowledge injection, with the structure illustrated in Figure 2. For each lexical relation $r \in \mathcal{R} \setminus \{\mathbf{RAN}\}$, we extract the *relation prototype* $\vec{r}_{proto}$ from the lexical knowledge base $\mathcal{KB}$ by averaging all the embedding offsets (i.e., $\vec{x}_i - \vec{y}_i$) of concept pairs (x_i, y_i) with relation r:

$$\vec{r}_{proto} = \frac{\sum_{(x_i,y_i,r_i)\in\mathcal{KB}}(\mathbf{1}(r_i = r) \cdot (\vec{x}_i - \vec{y}_i))}{\sum_{(x_i,y_i,r_i)\in\mathcal{KB}} \mathbf{1}(r_i = r)}$$

We use $\vec{x}_i - \vec{y}_i$ as features because the *Diff* model is effective for representing semantic relations (Fu et al. 2014; Vylomova et al. 2016; Wang, He, and Zhou 2019). Consider

the SRR Cell structure in Figure 2. Given the inputs $\vec{x}_i$, $\vec{y}_i$ and $\vec{r}_{proto}$, we compute the hidden states $\vec{U}_1$ and $\vec{U}_2$ by:

$$\vec{U}_1 = \tanh((\vec{x}_i \oplus \vec{r}_{proto}) \cdot \mathbf{W}_1 + \vec{b}_1)$$

$$\vec{U}_2 = \tanh((\vec{y}_i \oplus \vec{r}_{proto}) \cdot \mathbf{W}_2 + \vec{b}_2)$$

where $\mathbf{W}_1 \in \mathbb{R}^{2d \times d}$, $\mathbf{W}_2 \in \mathbb{R}^{2d \times d}$, $\vec{b}_1 \in \mathbb{R}^d$ and $\vec{b}_2 \in \mathbb{R}^d$ are the weights and biases of these hidden states. This can be interpreted as *inferring the embeddings of relation objects or subjects*, given the relation prototype and subjects/objects as inputs, similar to knowledge graph completion (Wang et al. 2017). Next, we compute the offsets $\vec{U}_1 - \vec{y}_i$ and $\vec{U}_2 - \vec{x}_i$ and two new d-dimensional hidden states $\vec{U}_3$ and $\vec{U}_4$, with $\mathbf{W}_3 \in \mathbb{R}^{d \times d}$, $\mathbf{W}_4 \in \mathbb{R}^{d \times d}$, $\vec{b}_3 \in \mathbb{R}^d$ and $\vec{b}_4 \in \mathbb{R}^d$ as learnable parameters. $\vec{U}_3$ and $\vec{U}_4$ are defined as:

$$\vec{U}_3 = \tanh((\vec{U}_1 - \vec{y}_i) \cdot \mathbf{W}_3 + \vec{b}_3)$$

$$\vec{U}_4 = \tanh((\vec{U}_2 - \vec{x}_i) \cdot \mathbf{W}_4 + \vec{b}_4)$$

We can see that if x_i and y_i actually have the lexical relation r, $\vec{U}_3$ and $\vec{U}_4$ should be good indicators of the existence of such relations. For example, one way to interpret $\vec{U}_1$ and $\vec{U}_3$ is that $\vec{U}_1$ tries to infer the relation object given $\vec{x}_i$ and $\vec{r}_{proto}$ as inputs, and $\vec{U}_3$ makes the judgment by comparing $\vec{U}_1$ and the true relation object embedding $\vec{y}_i$. Hence, the network learns whether $\vec{r}_{proto}$ is a good fit for the concept pair (x_i, y_i). The functionalities of $\vec{U}_2$ and $\vec{U}_4$ are similar, only with directions reversed. After that, we concatenate $\vec{U}_3$ and $\vec{U}_4$ as part of the inputs for the next layer.

Re-consider the entire network structure in Figure 1. For each concept pair (x_i, y_i), we compare $\vec{x}_i$ and $\vec{y}_i$ with all relation prototypes (treated as constants in the network). The results are represented by $2(|\mathcal{R}|-1)$ vectors of hidden states. After that, a dense layer and multiple output layers are connected. During the meta-learning stage, we train $|\mathcal{R}| - 1$ binary meta-classifiers to minimize $\mathcal{L}(\mathcal{T}_r)$ ($\mathcal{T}_r \in \mathcal{T}$), with meta-parameters updated. In the supervised learning stage, we discard all the output layers of meta-classifiers, and train the final LRC model. This is because the numbers of output units of meta-classifiers and the final classifier are different. The parameters of the last layer can not be re-used. We also need to note that additional *skip connections* between $\vec{x}_i$, $\vec{y}_i$ and the dense layer are added, in order to improve the effect of back propagation during training (He et al. 2016).

Discussion. Overall speaking, KEML employs a "divide-and-conquer" strategy for LRC. In the meta-learning step, each SRR Cell learns the semantics of single lexical relation type, and also handles the $\mathbf{RAN}$ problem. Hence, the supervised learning process of the relation classifier can be improved with better parameter initializations, as model parameters already have some knowledge about relation recognition. Unlike traditional meta-learning (Finn, Abbeel, and Levine 2017; Finn, Xu, and Levine 2018), KEML does not contain meta-testing steps (since LRC is not an N-way K-shot learning problem), but takes advantages of meta-learning as an intermediate step for the supervised training of the relation classifier afterwards.

Experiments

In this section, we conduct extensive experiments to evaluate KEML over multiple benchmark datasets, and compare it with state-of-the-arts to make the convincing conclusion.

Datasets and Experimental Settings

We employ Google's pre-trained BERT model[3] to initialize the parameters of LKB-BERT. The lexical knowledge base $\mathcal{KB}$ contains 16.7K relation triples [4]. Following Wang, He, and Zhou (2019), we use the five public benchmark datasets in English for multi-way classification of lexical relations to evaluate KEML, namely, K&H+N (Necsulescu et al. 2015), BLESS (Baroni and Lenci 2011), ROOT09 (Santus et al. 2016b), EVALution (Santus et al. 2015) and CogALex-V Subtask 2 (Santus et al. 2016a). Statistics of all the datasets are shown in Table 2. K&H+N, BLESS, ROOT09 and EVALution are partitioned into training, validation and testing sets, following the exact same settings as in Shwartz and Dagan (2016). The CogALex-V dataset has training and testing sets only, with no validation sets provided (Santus et al. 2016a). Hence, we randomly sample 80% of the training set to train the model, and use the rest for tuning.

The default hyper-parameter settings of KEML are as follows:[5] $N = |\mathcal{R}| - 1$, $\gamma = 1$ and $\alpha = \epsilon = 10^{-3}$. We use $tanh$ as the activation function, and Adam as the optimizer to train the neural network. All the model parameters are l_2-regularized, with the hyper-parameter $\lambda = 10^{-3}$. The batch size is set as 256. The dimension of hidden layers is set as the same of d (768 for the base BERT model). The number of parameters of the final neural classifier is around 7M to 24M, depending on the number of classes. The algorithms are implemented with TensorFlow and trained with NVIDIA Tesla P100 GPU. The training time of LKB-BERT for all datasets is about 1.5 hours. Afterwards, only a few minutes are required to train the model for each dataset. For evaluation, we use Precision, Recall and F1 as metrics, reported as the average of all the classes, weighted by the support.

General Experimental Results

We first evaluate KEML over K&H+N, BLESS, ROOT09 and EVALution. Since EVALution does not contain $\mathbf{RAN}$ relations, in the meta-learning process, we randomly sample relation triples from $\mathcal{D}$ that do not have the relation r, and take them as $\mathcal{S}_{\mathbf{RAN}}$. We manually tune the regularization hyper-parameter λ from 10^{-2} to 10^{-4} using the validation set (based on F1) and report the performance over the testing set. As for baselines, we consider traditional distributional models Concat (Baroni et al. 2012) and Diff (Weeds et al. 2014), pattern-based models NPB (Shwartz, Goldberg, and Dagan 2016), LexNET (Shwartz and Dagan

[3] We use the uncased, base version of BERT. See https://github.com/google-research/bert.

[4] To avoid data leakage, we have removed relation triples in $\mathcal{KB}$ such that at least one word (either the relation subject or object) overlaps with relation triples in validation and testing sets. This setting ensures our model can generate fair and unbiased results.

[5] We empirically set $N = |\mathcal{R}| - 1$ to ensure that in each iteration of meta-learning, each auxiliary task is learned once in average.

Method	K&H+N			BLESS			ROOT09			EVALution		
	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1
Concat	0.909	0.906	0.904	0.811	0.812	0.811	0.636	0.675	0.646	0.531	0.544	0.525
Diff	0.888	0.886	0.885	0.801	0.803	0.802	0.627	0.655	0.638	0.521	0.531	0.528
NPB	0.713	0.604	0.55	0.759	0.756	0.755	0.788	0.789	0.788	0.53	0.537	0.503
NPB+Aug	-	-	0.897	-	-	0.842	-	-	0.778	-	-	0.489
LexNET	0.985	0.986	0.985	0.894	0.893	0.893	0.813	0.814	0.813	0.601	0.607	0.6
LexNET+Aug	-	-	0.970	-	-	0.927	-	-	0.806	-	-	0.545
SphereRE	0.990	0.989	0.990	0.938	0.938	0.938	0.860	0.862	0.861	0.62	0.621	0.62
LKB-BERT	0.981	0.982	0.981	**0.939**	0.936	0.937	**0.863**	**0.864**	**0.863**	**0.638**	**0.645**	**0.639**
KEML-S	0.984	0.983	0.984	**0.942**	**0.940**	**0.941**	**0.877**	**0.871**	**0.873**	**0.649**	**0.651**	**0.644**
KEML	**0.993**	**0.993**	**0.993**	**0.944**	**0.943**	**0.944**	**0.878**	**0.877**	**0.878**	**0.663**	**0.660**	**0.660**

Table 1: LRC results over four benchmark datasets in terms of Precision, Recall and F1.

Relation	K&H+N	BLESS	ROOT09	EVAL.	Cog.
Antonym	-	-	-	1600	601
Attribute	-	2731	-	1297	-
Co-hyponym	25796	3565	3200	-	-
Event	-	3824	-	-	-
Holonym	-	-	-	544	-
Hypernym	4292	1337	3190	1880	637
Meronym	1043	2943	-	654	387
Random	26378	12146	6372	-	5287
Substance Meronym	-	-	-	317	-
Synonym	-	-	-	1086	402
Total	57509	26546	12762	7378	7314

Table 2: Dataset statistics. Relation names in all the datasets have been mapped to relation names in WordNet. "EVAL." and "Cog." are short for "EVALution" and "CogALex".

2016), NPB+Aug and LexNET+Aug (Washio and Kato 2018a), and the state-of-the-art model SphereRE (Wang, He, and Zhou 2019). We refer readers to the following papers (Shwartz and Dagan 2016; Washio and Kato 2018a; Wang, He, and Zhou 2019) for the detailed descriptions of these baselines. Additionally, we implement two variants of our approach: i) LKB-BERT (using trained concept representations to predict lexical relations) and ii) KEML-S (KEML without the meta-learning stage). The results of KEML and all the baselines are summarized in Table 1.

As shown, KEML outperforms all baselines, especially over BLESS, ROOT09 and EVALution. As for K&H+N, KEML produces a slightly better F1 score (0.3%) than the strongest baseline SphereRE (Wang, He, and Zhou 2019). A probable cause is that K&H+N is an "easy" dataset (99% F1 by SphereRE), leaving little room for improvement. Comparing KEML against LKB-BERT and KEML-S, we can conclude that, the knowledge enrichment technique for concept representation learning and meta-learning are highly beneficial for accurate prediction of lexical relations.

Results of CogALex-V Shared Task

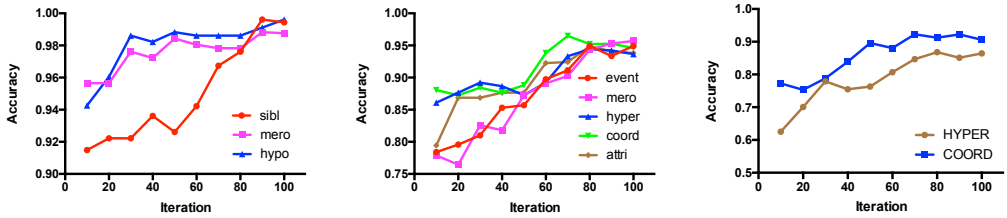
We evaluate KEML over the CogALex-V Shared Task (Subtask 2) (Santus et al. 2016a). This dataset is most challenging as it contains a large number of random word pairs and disables "lexical memorization". The organizer requires participants to discard the results of the random class from average, and report the F1 scores for each type of lexical relations. We consider two top systems reported in this task (i.e., GHHH (Attia et al. 2016) and LexNET (Shwartz and Dagan 2016)), as well as two recent models that have been evaluated over the shared task (i.e., STM (Glavas and Vulic 2018) and SphereRE (Wang, He, and Zhou 2019)) as strong competitors. Because the training set contains an overwhelming number of random word pairs, during the training process of KEML-S and KEML, we randomly discard 70% (manually tuned) of the random pairs in each epoch (which further improves the performance of KEML-S and KEML by 1.8% and 2.2%, in terms of overall F1). Results are reported in Table 3, showing that KEML achieves the highest F1 score of 50.0%. It also has highest scores on three types of lexical relations: synonymy (SYN), hypernymy (HYP) and meronymy (MER).

Method	SYN	ANT	HYP	MER	All
GHHH	0.204	0.448	0.491	0.497	0.423
LexNET	0.297	0.425	0.526	0.493	0.445
STM	0.221	**0.504**	0.498	0.504	0.453
SphereRE	0.286	0.479	0.538	0.539	0.471
LKB-BERT	0.281	0.470	0.532	0.530	0.464
KEML-S	0.276	0.470	**0.542**	**0.631**	**0.485**
KEML	**0.292**	0.492	**0.547**	**0.652**	**0.500**

Table 3: LRC results for each lexical relation types over the CogALex-V shared task in terms of F1.

Detailed Analysis of KEML

To facilitate deeper understanding, we conduct additional experiments to analyze KEML's components. We first study how knowledge-enriched concept representation learning benefits LRC. We implement three models: LKB-BERT (Binary), LKB-BERT (Multi) and LKB-BERT (Full). LKB-BERT (Binary) and LKB-BERT (Multi) only fine-tune on single objective: $\mathcal{L}_{KB}^{(2)}$ and $\mathcal{L}_{KB}^{(1)}$, respectively. LKB-BERT (Full) is the full implementation, as described previously. We take the two concepts' representations as features ($\vec{x}_i$ and $\vec{y}_i$) to train relation classifiers for LRC and report the results over the validation sets. For fair comparison, we use neural networks with one hidden layer (with the dimension



(a) Dataset: K&H+N (b) Dataset: BLESS (c) Dataset: ROOT09

Figure 3: Accuracy of single relation prediction during the meta-learning process (best viewed in color).

Dataset	Binary	Multi	Full
K&H+N	0.964	0.972	**0.983**
BLESS	0.921	0.929	**0.939**
ROOT09	0.854	0.861	**0.863**
EVALution	0.630	0.632	**0.641**
CogALex-V	0.464	0.467	**0.472**

Table 4: LRC results using concept embeddings generated by LKB-BERT and variants in terms of F1.

Concept Pairs	Predicted	True
(turtle, frog)	Synonym	Co-hyponym
(draw, pull)	Random	Synonym
(bowl, glass)	Co-hyponymy	Meronymy
(cannon, warrior)	Synonym	Random
(affection, healthy)	Co-hyponym	Attribute

Table 5: Cases of prediction errors.

Dataset: ROOT09

Prediction↓ True→	Co-hyponym	Hypernym	Random
Co-hyponym	*83.8%*	**8.2%**	**7.2%**
Hypernym	10.2%	*86.5%*	2.4%
Random	**6.0%**	5.3%	*90.4%*

Dataset: K+H&N

Prediction↓ True→	Co-hyponym	Hypernym	Meronym	Random
Co-hyponym	*99.4%*	**1.8%**	**1.0%**	0.2%
Hypernym	0.2%	*97.5%*	0.3%	0.1%
Meronym	0.1%	0.2%	*96.5%*	0.1%
Random	0.3%	0.5%	**2.2%**	*99.6%*

Figure 4: Inter-class prediction error analysis on ROOT09 and K+H&N. Top-3 error categories are marked in bold.

d, and the activation function $tanh$) as classifiers in all the experiments and present the results in Table 4. We can see that the improvement of using $\mathcal{L}_{KB} = \mathcal{L}^{(2)}_{KB} + \mathcal{L}^{(1)}_{KB}$ as the objective function is consistent across all the datasets (which ranges from 0.8% to 1.9% in terms of F1). Particularly, LKB-BERT outperforms the strongest baseline SphereRE in a few cases (for example, the dataset ROOT09, as shown in Table 1 and Table 4). Hence, LKB-BERT is capable of encoding knowledge in patterns and lexical knowledge bases to represent the semantics of lexical relations.

Next, we look into the meta-learning process in KEML. We test whether SRR Cells can distinguish a specific type of lexical relations from random concept pairs. In each iteration of meta-learning, we sample another batch of positive and negative samples from $\mathcal{D}$ and compute the accuracy of single relation recognition. Figure 3 illustrates how accuracy changes through time in K&H+N, BLESS and ROOT09. Within 100 iterations, our models can achieve desirable performance efficiently, achieving good parameter initializations for LRC. This experiment also explains why KEML produces better results than KEML-S.

Error Analysis

We analyze prediction errors produced by KEML. Because the inputs of our task are very simple and the interpretation of deep neural language models is still challenging, the error analysis process is rather difficult. Here, we analyze the causes of errors from a linguistic point of view, with some cases presented in Table 3. As seen, some types of lexical relations are very similar in semantics. For instance, concept pairs with the *synonymy* relation and the *co-hyponymy* relation are usually mapped similar positions in the embedding space. Hence, it is difficult for models to distinguish the differences between the two relations without rich contextual information available. Another problem is that some of the relations are "blurry" in semantics, making KEML hard to discriminate between these relations and random relations.

Additionally, we analyze the percentages of testing instances of each class being mis-classified to other classes. The results over ROOT09 and K+H&N are shown in Figure 4. Take K+H&N as an example. Although the prediction results are already highly accurate, for classes with few training instances (e.g., meronym), there is still room for further improvement, which will be left as future work.

Conclusion and Future Work

In this paper, we present the Knowledge-Enriched Meta-Learning (KEML) framework to distinguish different lexical relations. Experimental results confirm that KEML outperforms state-of-the-art approaches. Future work includes: i) improving relation representation learning with deep neural language models (Vylomova et al. 2016; Anke, Schockaert, and Wanner 2019; Camacho-Collados et al. 2019); ii) integrating richer linguistic and commonsense knowledge into KEML; and iii) applying KEML to downstream tasks such as taxonomy learning (Bordea, Lefever, and Buitelaar 2016; Jurgens and Pilehvar 2016; Wang et al. 2019).

Acknowledgements

This work is partially supported by the National Key Research and Development Program of China under Grant No. 2016YFB1000904. M. Qiu is partially funded by China Postdoctoral Science Foundation (No. 2019M652038).

References

Anke, L. E.; Schockaert, S.; and Wanner, L. 2019. Collocation Classification with Unsupervised Relation Vectors. In *ACL*, 5765–5772.

Attia, M.; Maharjan, S.; Samih, Y.; Kallmeyer, L.; and Solorio, T. 2016. CogALex-V Shared Task: GHHH - Detecting Semantic Relations via Word Embeddings. In *CogALex@COLING*, 86–91.

Baroni, M.; Bernardi, R.; Do, N.; and Shan, C. 2012. Entailment above the word level in distributional semantics. In *EACL*, 23–32.

Baroni, M.; and Lenci, A. 2011. How we BLESSed distributional semantic evaluation. In *GEMS Workshop*.

Bordea, G.; Lefever, E.; and Buitelaar, P. 2016. SemEval-2016 Task 13: Taxonomy Extraction Evaluation (TExEval-2). In *SemEval@NAACL-HLT*, 1081–1091.

Camacho-Collados, J.; Anke, L. E.; Jameel, S.; and Schockaert, S. 2019. A Latent Variable Model for Learning Distributional Relation Vectors. In Kraus, S., ed., *IJCAI*, 4911–4917.

Camacho-Collados, J.; Anke, L. E.; and Schockaert, S. 2019. Relational Word Embeddings. In *ACL*, 3286–3296.

Chen, M.; Zhang, W.; Zhang, W.; Chen, Q.; and Chen, H. 2019. Meta Relational Learning for Few-Shot Link Prediction in Knowledge Graphs. In *EMNLP-IJCNLP*, 4216–4225.

Cordeiro, S.; Ramisch, C.; Idiart, M.; and Villavicencio, A. 2016. Predicting the Compositionality of Nominal Compounds: Giving Word Embeddings a Hard Time. In *ACL*, 1986–1997.

Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J. G.; Le, Q. V.; and Salakhutdinov, R. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *ACL*, 2978–2988.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*, 4171–4186.

Dou, Z.; Yu, K.; and Anastasopoulos, A. 2019. Investigating Meta-Learning Algorithms for Low-Resource Natural Language Understanding Tasks. In *EMNLP-IJCNLP*, 1192–1197.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *ICML*, volume 70, 1126–1135.

Finn, C.; Xu, K.; and Levine, S. 2018. Probabilistic Model-Agnostic Meta-Learning. In *NeurIPS*, 9537–9548.

Fu, R.; Guo, J.; Qin, B.; Che, W.; Wang, H.; and Liu, T. 2014. Learning Semantic Hierarchies via Word Embeddings. In *ACL*, 1199–1209.

Glavas, G.; and Vulic, I. 2018. Discriminating between Lexico-Semantic Relations with the Specialization Tensor Model. In *NAACL*, 181–187.

Guo, D.; Tang, D.; Duan, N.; Zhou, M.; and Yin, J. 2019. Coupling Retrieval and Meta-Learning for Context-Dependent Semantic Parsing. In *ACL*, 855–866.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*, 770–778.

Hearst, M. A. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *COLING*, 539–545.

Joshi, M.; Choi, E.; Levy, O.; Weld, D. S.; and Zettlemoyer, L. 2019. pair2vec: Compositional Word-Pair Embeddings for Cross-Sentence Inference. In *NAACL*, 3597–3608.

Jurgens, D.; and Pilehvar, M. T. 2016. SemEval-2016 Task 14: Semantic Taxonomy Enrichment. In *SemEval@NAACL-HLT*, 1092–1102.

Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; and Soricut, R. 2019. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *CoRR* abs/1909.11942.

Le, M.; Roller, S.; Papaxanthos, L.; Kiela, D.; and Nickel, M. 2019. Inferring Concept Hierarchies from Text Corpora via Hyperbolic Embeddings. In *ACL*, 3231–3241.

Levy, O.; Remus, S.; Biemann, C.; and Dagan, I. 2015. Do Supervised Distributional Methods Really Learn Lexical Inference Relations? In *NAACL*, 970–976.

Liu, L.; Ren, X.; Zhu, Q.; Zhi, S.; Gui, H.; Ji, H.; and Han, J. 2017. Heterogeneous Supervision for Relation Extraction: A Representation Learning Approach. In *EMNLP*, 46–56.

Liu, W.; Zhou, P.; Zhao, Z.; Wang, Z.; Ju, Q.; Deng, H.; and Wang, P. 2019. K-BERT: Enabling Language Representation with Knowledge Graph. *CoRR* abs/1909.07606.

Madotto, A.; Lin, Z.; Wu, C.; and Fung, P. 2019. Personalizing Dialogue Agents via Meta-Learning. In *ACL*, 5454–5459.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient Estimation of Word Representations in Vector Space. In *ICLR*.

Miller, G. A. 1995. WordNet: A Lexical Database for English. *Commun. ACM* 38(11): 39–41. doi:10.1145/219717.219748.

Mrksic, N.; Vulic, I.; Séaghdha, D. Ó.; Leviant, I.; Reichart, R.; Gasic, M.; Korhonen, A.; and Young, S. J. 2017. Semantic Specialization of Distributional Word Vector Spaces using Monolingual and Cross-Lingual Constraints. *TACL* 5: 309–324.

Necsulescu, S.; Mendes, S.; Jurgens, D.; Bel, N.; and Navigli, R. 2015. Reading Between the Lines: Overcoming Data Sparsity for Accurate Classification of Lexical Relationships. In *SEM, 182–192.

Nguyen, K. A.; Schulte im Walde, S.; and Vu, N. T. 2016. Integrating Distributional Lexical Contrast into Word Embeddings for Antonym-Synonym Distinction. In *ACL*.

Nguyen, K. A.; Schulte im Walde, S.; and Vu, N. T. 2017. Distinguishing Antonyms and Synonyms in a Pattern-based Neural Network. In *EACL*, 76–85.

Nichol, A.; Achiam, J.; and Schulman, J. 2018. On First-Order Meta-Learning Algorithms. *CoRR* abs/1803.02999.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep Contextualized Word Representations. In *NAACL*, 2227–2237.

Ponti, E. M.; Vulic, I.; Glavas, G.; Reichart, R.; and Korhonen, A. 2019. Cross-lingual Semantic Specialization via Lexical Relation Induction. In *EMNLP-IJCNLP*, 2206–2217.

Qiu, X.; Sun, T.; Xu, Y.; Shao, Y.; Dai, N.; and Huang, X. 2020. Pre-trained Models for Natural Language Processing: A Survey. *CoRR* abs/2003.08271. URL https://arxiv.org/abs/2003.08271.

Roller, S.; and Erk, K. 2016. Relations such as Hypernymy: Identifying and Exploiting Hearst Patterns in Distributional Vectors for Lexical Entailment. In *EMNLP*, 2163–2172.

Roller, S.; Kiela, D.; and Nickel, M. 2018. Hearst Patterns Revisited: Automatic Hypernym Detection from Large Text Corpora. In *ACL*, 358–363.

Santus, E.; Gladkova, A.; Evert, S.; and Lenci, A. 2016a. The CogALex-V Shared Task on the Corpus-Based Identification of Semantic Relations. In *CogALex@COLING*, 69–79.

Santus, E.; Lenci, A.; Chiu, T.; Lu, Q.; and Huang, C. 2016b. Nine Features in a Random Forest to Learn Taxonomical Semantic Relations. In *LREC*.

Santus, E.; Yung, F.; Lenci, A.; and Huang, C. 2015. EVALution 1.0: an Evolving Semantic Dataset for Training and Evaluation of Distributional Semantic Models. In *LDL@ACL-IJCNLP*, 64–69.

Shen, J.; Wu, Z.; Lei, D.; Zhang, C.; Ren, X.; Vanni, M. T.; Sadler, B. M.; and Han, J. 2018. HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion. In *KDD*, 2180–2189.

Shwartz, V.; and Dagan, I. 2016. Path-based vs. Distributional Information in Recognizing Lexical Semantic Relations. In *CogALex@COLING*, 24–29.

Shwartz, V.; Goldberg, Y.; and Dagan, I. 2016. Improving Hypernymy Detection with an Integrated Path-based and Distributional Method. In *ACL*.

Thompson, B.; Knowles, R.; Zhang, X.; Khayrallah, H.; Duh, K.; and Koehn, P. 2019. HABLex: Human Annotated Bilingual Lexicons for Experiments in Machine Translation. In *EMNLP-IJCNLP*, 1382–1387.

Vanschoren, J. 2018. Meta-Learning: A Survey. *CoRR* abs/1810.03548.

Vylomova, E.; Rimell, L.; Cohn, T.; and Baldwin, T. 2016. Take and Took, Gaggle and Goose, Book and Read: Evaluating the Utility of Vector Differences for Lexical Relation Learning. In *ACL*, 1671–1682.

Wang, C.; Fan, Y.; He, X.; and Zhou, A. 2019. Predicting hypernym-hyponym relations for Chinese taxonomy learning. *Knowl. Inf. Syst.* 58(3): 585–610.

Wang, C.; He, X.; and Zhou, A. 2017. A Short Survey on Taxonomy Learning from Text Corpora: Issues, Resources and Recent Advances. In *EMNLP*, 1190–1203.

Wang, C.; He, X.; and Zhou, A. 2019. SphereRE: Distinguishing Lexical Relations with Hyperspherical Relation Embeddings. In *ACL*, 1727–1737.

Wang, Q.; Mao, Z.; Wang, B.; and Guo, L. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Trans. Knowl. Data Eng.* 29(12): 2724–2743. doi:10.1109/TKDE.2017.2754499.

Washio, K.; and Kato, T. 2018a. Filling Missing Paths: Modeling Co-occurrences of Word Pairs and Dependency Paths for Recognizing Lexical Semantic Relations. In *NAACL*, 1123–1133.

Washio, K.; and Kato, T. 2018b. Neural Latent Relational Analysis to Capture Lexical Semantic Relations in a Vector Space. In *EMNLP*, 594–600.

Weeds, J.; Clarke, D.; Reffin, J.; Weir, D. J.; and Keller, B. 2014. Learning to Distinguish Hypernyms and Co-Hyponyms. In *COLING*, 2249–2259.

Wu, S.; and He, Y. 2019. Enriching Pre-trained Language Model with Entity Information for Relation Classification. In *CIKM*, 2361–2364.

Wu, W.; Li, H.; Wang, H.; and Zhu, K. Q. 2012. Probase: a probabilistic taxonomy for text understanding. In *SIGMOD*, 481–492.

Xiong, W.; Du, J.; Wang, W. Y.; and Stoyanov, V. 2020. Pretrained Encyclopedia: Weakly Supervised Knowledge-Pretrained Language Model. In *ICLR*.

Yang, S.; Zou, L.; Wang, Z.; Yan, J.; and Wen, J. 2017. Efficiently Answering Technical Questions - A Knowledge Graph Approach. In *AAAI*, 3111–3118.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J. G.; Salakhutdinov, R.; and Le, Q. V. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *NeurIPS*, 5754–5764.

Yao, L.; Mao, C.; and Luo, Y. 2019. KG-BERT: BERT for Knowledge Graph Completion. *CoRR* abs/1909.03193.

Zhang, Z.; Han, X.; Liu, Z.; Jiang, X.; Sun, M.; and Liu, Q. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In *ACL*, 1441–1451.