Revisiting Text and Knowledge Graph Joint Embeddings: The Amount of Shared Information Matters!

Paolo Rosso
eXascale Infolab
University of Fribourg
Fribourg, Switzerland
paolo.rosso@unifr.ch

Dingqi Yang*
eXascale Infolab
University of Fribourg
Fribourg, Switzerland
dingqi.yang@unifr.ch

Philippe Cudré-Mauroux
eXascale Infolab
University of Fribourg
Fribourg, Switzerland
pcm@unifr.ch

***Abstract*—Jointly learning embeddings from text and a Knowledge Graph benefits both word and entity/relation embeddings by taking advantage of both large-scale unstructured content (text) and high-quality structured data (the Knowledge Graph). Current techniques leverage anchors to associate entities in the Knowledge Graph to corresponding words in the text corpus; these anchors are then used to generate additional learning samples during the embedding learning process. However, we show in this paper that such techniques yield suboptimal results, as they fail to control the amount of shared information between the two data sources during the joint learning process. Moreover, the additional learning samples often incur significant computational overhead. Aiming at releasing the power of such joint embeddings, we propose JOINER, a new joint text and Knowledge Graph embedding method using regularization. JOINER not only preserves co-occurrence between words in a text corpus and relations between entities in a Knowledge Graph, it also provides the flexibility to control the amount of information shared between the two data sources via regularization. Our method does not generate additional learning samples, which makes it computationally efficient. Our extensive empirical evaluation on real datasets shows the superiority of JOINER across different evaluation tasks, including analogical reasoning, link prediction, and relation extraction. Compared to state-of-the-art techniques generating additional learning samples from a set of anchors, our method yields better results (with up to 4.3% absolute improvement) and significantly less computational overhead (76% less learning time overhead).**

***Index Terms*—Word embeddings, Knowledge Graph embeddings, regularization**

I. Introduction

Jointly learning embeddings from both text and Knowledge Graphs (KGs) has been shown to combine the advantages from both *large-scale* unstructured data (text) and *high-quality* structured data (KGs). On one hand, word embeddings techniques, represent words in unstructured text as real-valued vectors, which can be efficiently used as features by various downstream applications such as document classification [1], named entity recognition [2], or sentiment analysis [3]. Skip-Gram [4], [5], for instance, is a well-known word embedding technique that can efficiently learn word embeddings by preserving the co-occurrence between words in a context window in a large text corpus, hence implicitly capturing semantic relations between words (e.g., *USA* $-$ *dollar* $\approx$ *Japan* $-$ *yen*). Although word embedding techniques can efficiently capture the semantics of words from large-scale unstructured text, they typically cannot model the nature of the *relation* between pairs of words, e.g., they cannot infer the *currency* relation between *Japan* and *yen*.

Knowledge Graphs (KGs), on the other hand, contain high-quality structured data where entities are connected via relations. More specifically, a typical KG, such as Freebase [6], Google's Knowledge Graph [7] or Wikidata [8], can be represented as a set of facts; each fact is a triplet *head, relation, tail* or (h, r, t) for short, such as *Japan* (head) *hasCurrency* (relation) *yen* (tail). To efficiently exploit KGs in practice, KG embeddings [9] have been proposed to represent entities as vectors, and relations as operations, while still preserving the relations between entities. A typical KG embedding technique is TransE [10]. It models a relation as a vector-plus operation and hence preserves the relations between entities as $h+r \approx t$. By doing so, it preserves the reasoning ability of a KG, i.e., a new fact can be asserted by evaluating $||h + r - t||$. Although the high-quality and structured data present in KGs have been widely used to power various applications, ranging from question answering [11] to query expansion [12], they are known to suffer from incompleteness (i.e., they are missing many/most entities and facts) [10], [13]. For example, 93.8% of persons from Freebase have no *place of birth*, while 78.5% have no *nationality* [14].

In the current literature, the most popular scheme to jointly learn embeddings from text and KGs [13], [15]–[17] is to define one or several types of anchors that link entities in a KG and words in a text corpus, such as entity names (labels) appearing in text or entities associated to Wikipedia pages. Subsequently, these anchors are used to generate additional learning samples during the embedding learning process. For example, when an entity's name (label) appears in text, beyond

*Corresponding author

the co-occurrence between words, a joint embedding model additionally learns from the co-occurrence of the entity and its surrounding words [13]. Although such a joint learning scheme has shown the advantage of combining text and KGs, we still raise the following question: *Does this scheme optimally combine the advantages from the two data sources?*

To answer this question, we started by reviewing the aforementioned joint learning process. More precisely, the heuristically-defined anchors used by this scheme uniquely determine the additionally-generated learning samples, and thus implicitly control the extend to which the word and KG embeddings are *jointly* learnt from the two data sources. Subsequently, it fails to provide the flexibility to control *how much information is actually shared between the two data sources in the embedding learning process*, which in many cases leads to suboptimal results (as we show below in our experiments). Even though one can combine different heuristics to generate different sets of anchors, it still gives very limited flexibility to control the *joint* learning process (see the baseline JointAS in our experiments for more details). Moreover, this joint learning scheme also suffers from an inefficiency issue, as the additionally generated learning samples require extra computation and learning time.

Aiming at releasing the power of joint text and KG embeddings, we propose *JOINER*, a novel Joint text and KG Embedding learning method via Regularization. JOINER not only preserves the co-occurrence between words in a text corpus and the relation between entities in a KG, but also provides the flexibility to smoothly control the amount of information shared between the two data sources in the joint embedding learning process using regularization. Specifically, we insert a regularization term in both the text and the KG embedding models to minimize the distance between a word (from a text corpus) and an entity (from a KG) when the two are linked by an anchor (e.g., entity names appearing in text). Subsequently, we are able to smoothly control via a regularization parameter the extent to which the word and the KG embeddings are *jointly* learnt from the two sources. Moreover, without additionally generated learning samples, JOINER incurs significantly less computation overhead compared to classical joint embedding learning methods. We conduct a thorough evaluation of JOINER on three evaluation tasks (analogical reasoning, link prediction and relation extraction) using Freebase, Wikipedia and New York Times corpora. Our results show the superiority of JOINER. In particular, compared to the state-of-the-art technique generating additional learning samples from a set of anchors [13], our method yields better performance (with a 1.4%-4.3% improvement on different evaluation tasks), and 76% less computational overhead. Moreover, our results also show that the amount of shared information matters across different evaluation tasks.

II. Related work

Word embedding techniques learn vector representations of words from a text corpus by preserving the semantic similarity between words [4], [18]–[20]. For example, the SkipGram model [4] learns word embeddings by maximizing the co-occurrence probability of words in a text corpus. The resulting embeddings are able to capture semantic and syntactic relations between words. By incorporating a negative sampling technique and using asychronized stochastic gradient descent to speedup the embedding learning process [5], the SkipGram model has been shown to be able to scale up to a large corpus in practice. Although word embedding models are able to efficiently capture semantic and syntactic relations between words from a large-scale corpus, they cannot extract explicit relations between pairs of words.

Knowledge Graph embedding techniques learn vector representations of entities and relations in a KG by preserving the relations between entities and thus maintaining the reasoning ability of the KG [9], [21]. The existing KG embedding techniques can be classified into two categories, i.e., translational distance models and semantic matching models. Translational distance models exploit distance-based scoring functions to create the embeddings. One representative model is TransE [10], which creates embeddings from triplet (h, r, t) such that the relation between the head and tail entities are preserved as $h + r \approx t$. Several works further improve TransE to capture complex KG structures, such as multi-mapping relations (one-to-many, many-to-one, or many-to-many), using more sophisticated scoring function involving relation-specific hyperplanes [22] or spaces [23], [24], for example. Ebisu et. al. [25] employ the same principle applied in TransE to a Lie group, which is a compact space representation. Semantic matching models, on the other hand, exploit similarity-based scoring functions. One typical model is RESCAL [26]. It represents each entity as a vector and each relation as a matrix, and uses a bilinear function to model the relation between two entities. Several works also extend RESCAL by putting a specific focus on reducing the model complexity [27], by capturing asymmetric relations [28], or by modeling non-linear relations using neural networks [29]–[33]. Despite achieving good performance, these non-linear models are computationally expensive, non-transparent and poorly understood, as opposed to translational distance and semantic matching models. Although KG embeddings are able to capture specific relations between entities, they suffer from incompleteness, i.e., many triplets are missing from the KG.

Joint text and Knowledge Graph embeddings learn embeddings by combining the two data sources. The most common scheme to jointly learn text and KG embeddings is to define one or several types of anchors that link entities in a KG and words in a text corpus, which are then used to generate additional learning samples in the embedding learning process. One representative method is proposed by [13]. Specifically, based on a SkipGram-like text model and a TransE-like KG model, it jointly learns word and entity/relation embeddings.

It defines two types of anchors connecting words in a text corpus and entities in a KG: 1) entity names (labels) appearing in text and 2) entity-associated Wikipedia pages, and then generates additional learning sample from these anchors. For entity names appearing in text, additional learning samples are generated for the KG model by replacing the entity from each head-relation-tail triplet by the corresponding word (e.g., word-relation-tail if the head is an anchor from text). For entity-associated Wikipedia pages, additional learning samples are generated for the text model by replacing the word from each word pairs by the associated entity (e.g., word-entity). [15] extends this model by using another type of anchors, i.e., words appearing in the entity description from the KG. Following this joint learning scheme, many task-specific joint embedding methods have also been proposed. For example, [16] combines all the aforementioned types of anchors to jointly learn embeddings specifically for the entity disambiguation problem. [17] addresses the same problem considering not only entity names appearing in text as anchors, but also the relatedness between entities. [34] learns joint text and KG embeddings using entity name appearing in text as anchors, and also by inferring relations from text using distant supervision. However, all these methods do not provide the flexibility to control how much information is actually shared between the two data sources in the embedding learning process, and thus fail to optimally take advantage of both data sources. Moreover, the additional samples generated from the anchors create some significant overhead in terms of computations and learning time, in particular when a large number of additional samples being fed to a stochastic optimization process [35]. To address these issues, we propose JOINER to jointly learn text and KG embeddings via regularization. JOINER can flexibly control the amount of information shared between the two data sources in the embedding learning process with significantly less computational overhead.

In addition, there are some other embedding methods combining text and KGs focusing on improving one using the other, but they do not tackle the same problem as ours. On the one hand, KG embeddings can be improved using textual relations (i.e., full lexicalized dependency paths) [36] or sentence/paragraph embeddings [37]. On the other hand, word embeddings can be enhanced by considering explicit relations from semantic knowledge [38] or word concepts from KGs [39]. However, these methods output either word embeddings or entity/relation embeddings.

Finally, regularization techniques are widely used for joint learning from multiple data sources [40], [41]. To the best of our knowledge, for joint learning from text and KGs, only one recent work KADE [42] adopts regularization techniques to align entities with *documents* using heuristically defined relatedness between entities and documents, which differs from our goal (aligning entities with *words*). Meanwhile, it outputs document embeddings rather than word embeddings, and thus fails to support the analogy reasoning task.

III. *JOINER*

A. Model

Our proposed joint model is built on top of a text model for learning word embeddings and a KG model for learning entity/relation embeddings. In this paper, we adopt the same individual text and KG models as in [13], which is a representative joint text and KG embedding method generating additional learning samples from a set of anchors. Subsequently, *by comparing with [13] in the experiments, the superiority of using regularization when jointly learning the embeddings can be clearly verified*, as we discount the effect of the individual text and KG models. However, we also note that our joint embedding learning method using regularization is not limited to any specific text and KG model; it can incorporate more sophisticated text or KG models (more discussion on this point later). In the following, we start by describing the individual text and KG models, followed by our joint model.

Text model. Our text model learns word embeddings by capturing the co-occurrence of words observed in a text corpus $\mathcal{D}$. We adopt the same text model as in [13]. Specifically, let $\mathcal{V}$ denote the word vocabulary of the text corpus[1]. The conditional probability of a target word w appearing close to a context word v (within a context window of a certain length) is defined as follows:

$$p(w|v) = \frac{\exp\{s(\mathbf{w}, \mathbf{v})\}}{\sum_{v' \in V} \exp\{s(\mathbf{w}, \mathbf{v}')\}} \tag{1}$$

where $s(\mathbf{w}, \mathbf{v})$ is a scoring function evaluating the co-occurrence of two words w and v based on their embeddings. It is defined as $s(\mathbf{w}, \mathbf{v}) = b - \frac{1}{2}||\mathbf{w} - \mathbf{v}||^2$, where b is a constant margin used for better numerical stability in the learning process. Subsequently, the objective of the text model is to maximize the likelihood of the co-occurrence of pairs of words in the whole text corpus:

$$\mathcal{L}_T = \sum_{(w,v) \in \mathcal{C}} \#(w, v) \log p(w|v) \tag{2}$$

where $\mathcal{C}$ is the set of unique word pairs co-occurring in a context window of a certain length, and $\#(w, v)$ is the number of times (w, v) appears in the corpus $\mathcal{D}$.

Knowledge Graph model. Our KG model learns entity/relation embeddings by preserving the relations between entities. Specifically, a KG Δ consists of a set of triplets (h, r, t), $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$, where $\mathcal{E}$ and $\mathcal{R}$ refer to the entity and relation vocabularies, respectively. Similar to the text model, we define the conditional probability of observing h given (r, t) in the KG as follows:

$$p(h|r, t) = \frac{\exp\{s(\mathbf{h}, \mathbf{r}, \mathbf{t})\}}{\sum_{h' \in \mathcal{E}} \exp\{s(\mathbf{h}', \mathbf{r}, \mathbf{t})\}} \tag{3}$$

where $s(\mathbf{h}, \mathbf{r}, \mathbf{t})$ is a scoring function evaluating the correctness of the triplet (h, r, t) based on their embeddings. It is defined

[1]We pre-process the text corpus in order to detect common phrases (e.g., *new york*) such that the vocabulary $\mathcal{V}$ also contains these phrases. For the sake of simplicity, we use the term "word(s)" to refer these words/phrases in $\mathcal{V}$, if not specified otherwise.

as $s(\mathbf{h}, \mathbf{r}, \mathbf{t}) = b - \frac{1}{2} \cdot ||\mathbf{h} + \mathbf{r} - \mathbf{t}||^2$. In addition, $p(r|h,t)$ and $p(t|h,r)$ are defined in the same way as $p(h|r,t)$ with the corresponding normalization terms, respectively. Subsequently, the KG model maximizes the likelihood of observing all triplets from the KG:

$$\mathcal{L}_{KG} = \sum_{(h,r,t)\in\Delta} [\log p(h|r,t) + \log p(r|h,t) + \log p(t|h,r)] \tag{4}$$

Joint Text and KG model using regularization. Our joint model combines the above text and KG models via regularization. Specifically, let $\mathcal{A}$ denote a set of anchors, where each anchor connects a word from the text corpus and an entity from the KG. In this paper, we adopt two common types of anchors, 1) entity names appeared in text and 2) entity-associated Wikipedia pages, which are widely used in the current literature [13], [16]. First, entity names naturally link entities to words in the text corpus. For each entity $e \in \mathcal{E}$, if its name (label) w_e also appears in our word vocabulary, i.e., $w_e \in V$, we consider (e, w_e) as an anchor. Second, entity-associated Wikipedia pages also link entities to words. A Wikipedia (English) page is often associated with a unique entity in a KG (e.g., Freebase). Subsequently, a Wikipedia anchor (i.e., a word w with a hyperlink to the Wikipedia page) in the text actually links to the page's associated entity e_w; we also considers (e_w, w) as an anchor. Finally, both types of anchors are included in our set of anchors $\mathcal{A}$, and we do not distinguish them in $\mathcal{A}$. Note that we do not consider entity descriptions as anchors [15], as it suffers from a low coverage issue (entity descriptions are not always available in a KG).

Based on the set of anchors $\mathcal{A}$, we are then able to jointly learn the text and KG models not only by capturing the co-occurrence between words in a text corpus and the relation between entities in a KG, but also by minimizing the distance between a word and an entity when the two are linked by an anchor via regularization. To achieve this goal, we insert regularization terms in both the text and KG models to connect them by minimizing the Euclidean distance between a word and an entity if they are connected by an anchor in $\mathcal{A}$. Specifically, for the text model, we redefine the scoring function as follows:

$$s(\mathbf{w}, \mathbf{v}) = b - \frac{1}{2}(||\mathbf{w} - \mathbf{v}||^2 + \mathbb{1}_{(v,e_v)\in\mathcal{A}} \cdot \beta \cdot ||\mathbf{v} - \mathbf{e}_v||^2) \tag{5}$$

where $\mathbb{1}_{(v,e_v)\in\mathcal{A}}$ is an indicator function which is equal to 1 when (v, e_v) is an anchor in $\mathcal{A}$ and to 0 otherwise. β is a regularization parameter which defines the weight of the regularization term $||\mathbf{v} - \mathbf{e}_v||^2$ (more on β below). Using this scoring function for our text model, we are able to not only capture the word co-occurrence (i.e., $||\mathbf{w} - \mathbf{v}||^2$), but also to preserve the similarity between a word and its anchor entity (if any) via the regularization term. Subsequently, we redefine the scoring function for the KG model in a similar way:

$$s(\mathbf{h}, \mathbf{r}, \mathbf{t}) = b - \frac{1}{2}(||\mathbf{h} + \mathbf{r} - \mathbf{t}||^2 + \mathbb{1}_{(h,w_h)\in\mathcal{A}} \cdot \beta \cdot ||\mathbf{h} - \mathbf{w}_h||^2 + \mathbb{1}_{(t,w_t)\in\mathcal{A}} \cdot \beta \cdot ||\mathbf{t} - \mathbf{w}_t||^2) \tag{6}$$

where we insert two regularization terms for head and tail entities, respectively. This scoring function is able to not only capture the relations between entities (i.e., $||\mathbf{h} + \mathbf{r} - \mathbf{t}||^2$), but also preserves the similarity between an entity and its word anchor via the two regularization terms. We use the same regularization parameter β for the KG model as for the text model, as our regularization terms share the same formulation in both the text and KG models.

In our joint learning model, *the regularization parameter β plays a key role to provide the flexibility to control the amount of information shared between the two data sources.* More precisely, β actually defines the weight of the regularization term in the whole scoring function. A smaller value of β implies a lower importance of the regularization term in the scoring function; the learning process will learn less from the set of anchors, meaning that less information is actually shared between the two models in the learning process. In the extreme case where we set $\beta = 0$, our joint learning model degrades to two independent text and KG models. In practice, β needs to be appropriately set to obtain high-quality word and entity/relation embeddings. On one hand, a small value of β may lead to insufficient information sharing between the models in the learning process. On the other hand, too big a value may lead to putting too much emphasis on the set of anchors while insufficiently learning from the two data sources (text and KGs). We investigate the impact of β across different evaluation tasks in our experiments. Below, we present our joint learning process.

B. Joint Embedding Learning Process

Learning with negative sampling. In practice, the considerable size of the vocabularies $\mathcal{V}$ and $\mathcal{E}$ makes it difficult to compute the normalizer in $p(w|v)$ (Eq. 1) and $p(h|r,t)$ (Eq. 3) (also for $p(r|h,t)$ and $p(t|h,r)$). To overcome this issue, negative sampling techniques [5] can be adopted to simplify the objective functions $\mathcal{L}_T$ and $\mathcal{L}_{KG}$. Specifically, for a pair of words (w, v), we not only maximize the probability of their co-occurrence, i.e., $p((w,v) \in \mathcal{D}|\mathbf{w}, \mathbf{v}) = \sigma(s(\mathbf{w}, \mathbf{v}))$, but also maximize the probability of the word w and a randomly sampled negative word v_N not appearing in the corpus D, i.e., $p((w, v_N) \notin \mathcal{D}|\mathbf{w}, \mathbf{v}_N) = \sigma(-s(\mathbf{w}, \mathbf{v}_N))$, where $\sigma(\cdot)$ is the sigmoid function. In summary, for each pair of words (w, v), we maximize the following objective function:

$$\Theta_w = \log \sigma(s(\mathbf{w}, \mathbf{v})) + \gamma \mathbb{E}_{v_N}[\log \sigma(-s(\mathbf{w}, \mathbf{v}_N))] \tag{7}$$

where $\gamma \in \mathbb{Z}^+$ is the number of negative samples. By applying the same negative sampling technique to the KG model, instead of maximizing $p(h|r,t)$, we maximize the following objective function:

$$\Theta_h = \log \sigma(s(\mathbf{h}, \mathbf{r}, \mathbf{t})) + \gamma \mathbb{E}_{h_N}[\log \sigma(-s(\mathbf{h}_N, \mathbf{r}, \mathbf{t}))] \tag{8}$$

where h_N is a randomly sampled negative head entity. We also convert $p(r|h,t)$ and $p(t|h,r)$ in the same way:

$$\Theta_r = \log \sigma(s(\mathbf{h}, \mathbf{r}, \mathbf{t})) + \gamma \mathbb{E}_{r_N}[\log \sigma(-s(\mathbf{h}, \mathbf{r}_N, \mathbf{t}))] \tag{9}$$

Algorithm 1 JOINER

Require: A text corpus $\mathcal{D}$, a KG Δ, and the corresponding vocabulary of words, entities and relations ($\mathcal{V}$, $\mathcal{E}$ and $\mathcal{R}$, respectively)

1: Initialize word, entity and relation embedding vectors $\mathbf{w}$ ($w \in \mathcal{V}$), $\mathbf{e}$ ($e \in \mathcal{E}$) and $\mathbf{r}$ ($r \in \mathcal{R}$)
2: **repeat**
3: Sample a batch of word pairs D_{batch} from $\mathcal{D}$
4: **for** $(w, v) \in D_{batch}$ **do**
5: Update $\mathbf{w}, \mathbf{v}$ with the gradients of Θ_w (Eq.7)
6: **end for**
7: Sample a batch of triples from Δ_{batch} from Δ
8: **for** $(h, r, t) \in \Delta_{batch}$ **do**
9: Update $\mathbf{h}, \mathbf{r}, \mathbf{t}, \mathbf{h}_N$ with the gradients of Θ_h(Eq.8)
10: Update $\mathbf{h}, \mathbf{r}, \mathbf{t}, \mathbf{r}_N$ with the gradients of Θ_r(Eq.9)
11: Update $\mathbf{h}, \mathbf{r}, \mathbf{t}, \mathbf{t}_N$ with the gradients of Θ_t(Eq.10)
12: **end for**
13: **until** Convergence

$$\Theta_t = \log \sigma(s(\mathbf{h}, \mathbf{r}, \mathbf{t})) + \gamma \mathbb{E}_{t_N}[\log \sigma(-s(\mathbf{h}, \mathbf{r}, \mathbf{t}_N))] \quad (10)$$

Alternating joint learning process. Based on the above objective functions, our joint learning process alternates between the two data sources. As shown in Algorithm 1, after initializing word/entity/relation embeddings (line 1), we alternatively learn from a batch of word pairs sampled by scanning the text corpus (Line 3-6), and a batch of triplets randomly sampled from the KG (Line 7-12). The batch size is empirically set to 500; in practice, when setting it to a small value (much smaller than the size of the corpora), it has negligible impact on the results. Our learning process is performed using asynchronous stochastic gradient descent (ASGD) until convergence. In practice, we iterate both datasets multiple times to ensure the convergence (see experiment settings for more details).

C. Extensibility of JOINER

In this paper, our JOINER model combines a text model and a KG model, whose scoring functions are both defined using the Euclidean distance. Our regularization term is also defined using the Euclidean distance. Our model is not limited to these two text and KG models, however. It can be modified to incorporate other text and KG models, under the condition that the scoring functions of the two models are defined using the same distance metric (e.g., cosine distance), with which our regularization term should also be defined. In this study, we instantiate JOINER using the Euclidean distance to distinctly demonstrate its advantages over [13] which uses the *same* text *and* KG model as ours but generates additional learning samples for the joint learning process.

IV. Experimental Evaluation

We evaluate JOINER on three different tasks: analogical reasoning, link prediction in KGs, and relation extraction. Subsequently, we investigate the impact of the regularization parameter β, followed by a runtime evaluation. We start below by presenting our experiment settings.

A. Experiment Setting

Dataset. Similar to [13], we use the following public text corpora and KGs.

- *Text.* We use the English Wikipedia dump collected in July 2017. We filter out noisy pages (e.g., pages for disambiguation), and perform named entity recognition to detect common phrases, which are included in the vocabulary $\mathcal{V}$[1]. After removing rare words, our text corpus $\mathcal{D}$ contains 1,110,804,425 words in total, with 489,861 unique words.
- *Knowledge Graph.* We adopt Freebase in our experiments, which has been widely used by previous work [10], [13], [22], [24]. We extract facts (triplets) from the Freebase dump[2], and for each entity we take its label in English as its entity name for anchor extraction. We consider the top 200K frequent entities and retain all related facts. In the end, our KG contains 199,355 unique entities, 3,442 unique relations and 2,459,553 triplets. For the link prediction task, we randomly sample 20% of the triplets as test data, and the rest as training data. We choose this KG rather than the commonly used FB15k or WN18, because its large scale ensures sufficient anchors linking entities to words.

Based on the above datasets, we extract the set of anchors $\mathcal{A}$, which sum up to 98,812 unique anchors. This represents 20.17% of the unique words in $\mathcal{V}$ and 49.57% of the unique entities in $\mathcal{E}$. Moreover, we find that these anchors actually involve 2,301,519 triplets (93.57% of all triplets) in the KG, and 398,551,199 words (35.88% of all words) in our text corpus. These statistics imply that anchors tend to connect frequent words and entities rather than infrequent ones.

Baselines. We compare our method against a sizable collection of nine state-of-the-art techniques from three categories.

- Word embedding techniques:
  - **SkipGram** [4] is a popular word embedding model that maximizes the co-occurrence probability of a word and its context in a text corpus, where the scoring function is defined using the dot product, i.e., $s(\mathbf{w}, \mathbf{v}) = \mathbf{w} \cdot \mathbf{v}$. Our text model is equivalent to the SkipGram model when constraining the embedding norm to be $||\mathbf{w}|| = ||\mathbf{v}|| = 1$, as $\mathbf{w} \cdot \mathbf{v} = 1 - 0.5 \cdot \|\mathbf{w} - \mathbf{v}\|^2$. Note that we adopt the Euclidean distance in the scoring function of our text model (rather than using the SkipGram model), in order to be consistent with our KG model (whose scoring function is also defined using the Euclidean distance).
  - **GloVe** [20] is another word embedding techniques that learns directly from aggregated global word-word co-occurrence statistics from a text corpus. The scoring function is defined using the dot product, i.e., $s(\mathbf{w}, \mathbf{v}) = \mathbf{w}^\top \cdot \mathbf{v} + b_w + b_v$, where b_w and b_v are scalar biases for a word and its context.
- Knowledge graph embedding techniques:

[2]https://developers.google.com/freebase/

- **TransE** [10] is a popular KG embedding model that preserves the relation between two entities as $h+r \approx t$. We also use it as our KG model. However, its objective function is different from ours. Specifically, TransE minimizes a margin-based ranking objective function, i.e., $[d(\mathbf{h}+\mathbf{r},\mathbf{t})+b-d(\mathbf{h}_N+\mathbf{r},\mathbf{t}_N)]_+$, where $d(\cdot,\cdot)$ can either be the L_1 or L_2-norm. In our experiments, we test different margins b with both L_1 and L_2-norm, the optimal setting being $b=1$ with the L_1-norm for TransE.
- **TransH** [22] is a KG embedding technique that further extend TransE to better capture multi-mapping relations in a KG. Specifically, TransH introduces the idea of relation-specific hyperplanes, in which, for a given triplet (h,r,t), the embeddings of the entities $\mathbf{h}$ and $\mathbf{t}$ are projected into a relation hyperplane $\mathbf{w}_r$, and their projections are denoted as $\mathbf{h}_\perp=\mathbf{h}-\mathbf{w}_r^\top\mathbf{h}\mathbf{w}_r$ and $\mathbf{t}_\perp=\mathbf{t}-\mathbf{w}_r^\top\mathbf{t}\mathbf{w}_r$, respectively. Thus, $\mathbf{h}_\perp$ and $\mathbf{t}_\perp$, can be connected by a translation vector $\mathbf{v}_r$ on the hyperplane. Similar to TransE, the relation between two entities is preserved as $h_\perp+v_r\approx t_\perp$, and the objective function is defined as $[d(\mathbf{h}_\perp+\mathbf{v}_r,\mathbf{t}_\perp)+b-d(\mathbf{h}_{\perp N}+\mathbf{v}_r,\mathbf{t}_{\perp N})]_+$, where $d(\cdot,\cdot)$ can either be the L_1 or L_2-norm. In our experiments, we also test different margins b with both L_1 and L_2-norm, the optimal setting being $b=0.25$ with the L_1-norm for TransH.
- **TransD** [24] is a KG embedding technique that decomposes the projection matrix into a product of two vectors. Specifically, additionally mapping vectors $\mathbf{w}_h$, $\mathbf{w}_t\in\mathcal{R}^d$ and $\mathbf{w}_r\in\mathcal{R}^k$ are introduced along with the entity and relation representations $\mathbf{h}$, $\mathbf{t}\in\mathcal{R}^d$ and $\mathbf{r}\in\mathcal{R}^k$. The two projection matrices are defined as $\mathbf{M}_r^1=\mathbf{w}_r\mathbf{w}_h^\top+\mathbf{I}$, and $\mathbf{M}_r^2=\mathbf{w}_r\mathbf{w}_t^\top+\mathbf{I}$. These two projection matrices are then used to project the head and tail entities with $\mathbf{h}_\perp=\mathbf{M}_r^1\mathbf{h}$ and $\mathbf{t}_\perp=\mathbf{M}_r^2\mathbf{t}$. With the projected entities, the objective function is defined in the same way as in TransR. In our experiments, we also test different margins b with both L_1 and L_2-norm, the optimal setting being $b=1$ with the L_1-norm for TransD.
- **DistMult** [27] is an efficient semantic matching model using a bilinear scoring function that maps each entity to *k*-dimensional vector and each relation *r* to a diagonal matrix $\mathbf{R}_r^{k\times k}$. For a given triplet, the score is computed as $s(h,r,t)=\mathbf{h}^\top\mathbf{R}_r\mathbf{t}$. DistMult also minimizes a margin-based ranking objective function, i.e., $max\{s(h_N,r,t_N)-s(h,r,t)+b,0\}$. In our experiments, we also test different margins b, the optimal setting being $b=0.1$ for DistMult.

- Knowledge graph embedding techniques:
  - **JointAS** [13] is a representative joint text and KG embedding model using Additional learning Samples generated from two types of anchors. It uses the same text and KG models as ours. It defines two types of anchors that link words and entities and then generate additional learning samples from these anchors. First, for entity names appeared in text, additional learning samples are generated for the KG model by replacing the entity from each triplet (h,r,t) by the corresponding word, e.g., (w_h,r,t) if (h,w_h) is an anchor in $\mathcal{A}$. This type of anchors is called "AN". For entity-associated Wikipedia pages, additional learning samples are generated for the text model by replacing the word from each word pairs (w,v) by the associated entity, e.g., (w,e_v) if (e_v,v) is an anchor in $\mathcal{A}$. This type of anchors is called "AA". Its joint learning process alternates between the text, the KG and the two types of additionally generated learning samples. As suggested by the authors, we set the margin to its optimal value $b=7$. Note that according to the selected types of anchors, JointAS can be configured in three settings, i.e., (AN), (AA) and (AN+AA).

We exclude task-specific joint embedding methods such as [16], [17] from our baselines, as they involve task-specific heuristics, and cannot support different evaluation tasks. We also exclude KADE [42], because it aligns entities with documents and thus outputs document embeddings rather than word embeddings, failing to support the analogy reasoning task.

For our JOINER, we search the optimal regularization parameter β from 0.0001 to 0.1 on a log scale, and set the margin $b=8$. For all the methods, we set the embedding size to 100, the number of negative samples to 10, and the context window size for text models to 5. For all models involving the text corpus, we train embeddings with three epochs on the text corpus (for the joint models, the KG is simultaneously traversed multiple times using the alternating joint learning process). For a fair comparison, we also train TransE, TransH, TransD and DistMult by traversing the KG the same number of times as for the joint models. We train and test five models for each method and report the average results. The code for our model and datasets are made publicly available[3].

B. Task 1: Analogical Reasoning

The analogical reasoning task is widely used to evaluate word embeddings. It consists of a set of analogies, such as *USA→dollar: Japan→?* that require to predict the missing word (*yen* in this example). We use a popular analogy dataset provided by [5], which contains 19,544 word and 3,218 phrase analogies. To predict the missing word for each analogy (denoted as *A→B:C→?*), we compute the Euclidean distance between $\mathbf{C}+(\mathbf{B}-\mathbf{A})$ and all the word embeddings, and pick the word with the minimum Euclidean distance as the predicted word. We report the accuracy over the whole analogy dataset.

Table I shows the results. First, we observe that JOINER outperforms all baselines by achieving the highest accuracy (with a 2.8% improvement over the best performing baseline SkipGram). Here we also report the corresponding optimal regularization parameter $\beta=0.001$ (we will later show

[3]https://github.com/eXascaleInfolab/JOINER_code/

TABLE I
ANALOGICAL REASONING PERFORMANCE

Method	Accuracy (%)
SkipGram	56.3
GloVe	47.2
JointAS (AN)	38.9
JointAS (AA)	50.7
JointAS (AN+AA)	40.5
JOINER (β=0.001)	**57.9**

TABLE II
LINK PREDICTION PERFORMANCE

Method	Hits@10 Raw (%)		Hits@10 Filtered (%)	
	Head	Tail	Head	Tail
TransE	37.1	44.6	41.7	49.2
TransH	38.7	45.4	46.1	52.6
TransD	38.9	43.1	48.6	52.7
DistMult	35.7	36.8	44.5	45.4
JoinAS (AN)	41.4	**49.1**	53.9	60.8
JoinAS (AA)	40.1	46.3	52.4	58.0
JoinAS (AN+AA)	40.9	48.7	52.5	59.7
JOINER (β=0.001)	**42.2**	48.9	**56.2**	**62.1**

that the optimal β varies across different tasks). Second, the best-performing word embedding technique, SkipGram, surprisingly beats JointAS on all configurations, which is the opposite of the results from [13]. This is probably due to the fact that our text and KG datasets are more tightly connected by anchors than the dataset used in [13], in particular for the text corpus. More precisely, 35.88% of all words in our text corpus and 93.57% of all triplets in our KG are connected by anchors, while these two percentages are 2.76% and 40% in the dataset used by [13], respectively. Subsequently, on our dataset, JointAS generates too many additional learning samples which dominate the learning process, leading to degraded results. In contrast, our JOINER approach allows to control the extent to which the embeddings are jointly learnt from the two data sources, and can hence avoid learning too much from the anchors by setting the regularization parameter β accordingly. Finally, we observe that JointAS results are sensitive to the anchor configuration; AA is the best anchor type for the analogical reasoning task, while AN actually pollutes the results. Similar findings are also reported in [13]. However, choosing appropriate anchors is not straightforward for JointAS; we will see later that different types of anchors may have varying impact on different tasks.

C. Task 2: Link Prediction

The link prediction task is widely used for KG completion [10]. It suggests new triplet (h, r, t) with h or t missing. In other words, it predicts t given (h, r) or predicts h given (r, t). To implement this task, we follow the same evaluation protocol as in TransE [10]. More precisely, when predicting t given (h, r), we compute the scores for $||\mathbf{h} + \mathbf{r} - \mathbf{e}||$, where $e \in \mathcal{E}$; by ranking the scores in ascending order, we generate a predicted ranking list of entities. We then report $Hits@10$, which measures the percentage of the predictions whose top-10 entities contain the ground truth entity t, over all triplets in the test dataset. We call this setting "raw". Moreover, when predicting t given (h, r), other tail entities may co-exist in the KG, i.e., $T' = \{t' | t' \neq t, (h, r, t') \in \Delta\}$. In this case, ranking t' in front of t should not be counted as an error. To avoid this case, [10] suggests filtering out these entities (T') before generating the ranked list of entities. We call this setting "filtered". The above evaluation protocol also applies to predict h given (r, t).

Table II shows the results. First, we observe that JOINER achieves the best performance in most cases (except when predicting tail with the "raw" setting, JointAS (AN) is slightly better). In the "filtered" setting (which is more reasonable for this task), JOINER outperforms the best-performing baseline JointAS (AN) with 2.1% and 4.3% improvement in predicting tail and head, respectively. Second, we find that JointAS with all configurations achieves better results than TransE, TransH, TransD and DistMult, showing the effectiveness of joint embeddings for this task. Moreover, for JointAS, we find that the anchor configuration AN yields higher performance than AA on this task, while we observe the opposite (i.e., AA shows better results than AN) on the analogical reasoning task. This implies that JointAS is less robust to different tasks, as the selection of anchors has a strong impact on the results. In contrast, our JOINER is less sensitive to anchor selection, as optimal performance can be achieved by tuning the regularization parameter.

D. Task 3: Relation Extraction

Relation extraction finds relations between two entities from text [43]. Given two detected entities in a sequence of text, it assigns a specific relation (if any) to the pair of entities based on their contexts (features) in text. We adopt the same evaluation protocol as used by JointAS [13]. Specifically, we first use a basic relation extractor [44] to generate a set of candidate relations r_i for each pair of entities (h, t) with their estimated probability $Pr_{mintz}(r_i)$ (according to text features). Then, we compute the Euclidean norms for the corresponding candidate triplets based on the embeddings, and convert the Euclidean norms into probabilities using a softmax function, i.e., $Pr_{JOINER}(r_i) = 1 - softmax(||\mathbf{h} + \mathbf{r}_i - \mathbf{e}||)$. Finally, for each candidate relation, we linearly combine the two probabilities as $\alpha \cdot Pr_{mintz}(r_i) + (1-\alpha) \cdot Pr_{JOINER}(r_i)$ and pick the relation with the largest probability. Similar to [13], the optimal α is searched from 0 to 1 with a step of 0.025. In this task, we use a relation-labeled dataset NYT+FB [45], and randomly split it into 50% training and 50% test sets. The basic extractor is trained on the training data, while entity/relation embeddings are trained using our own Wikipedia+Freebase datasets. We report the average accuracy over the test dataset.

Figure 1 shows the results. First, we observe that compared to the basic extractor Mintz, using embeddings can significantly improve the relation extraction performance; a similar observation has also been reported by [43]. Second, compared to the KG embedding techniques TransE, TransH, TransD

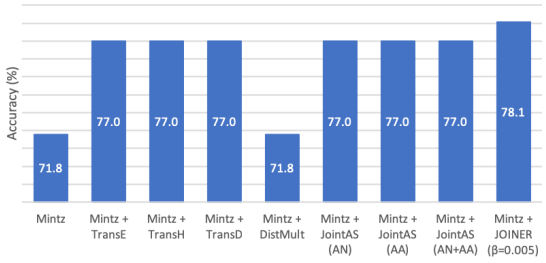
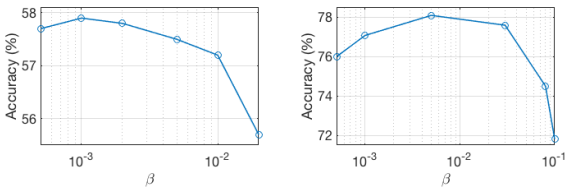


Fig. 1. Relation extraction performance



(a) Analogical reasoning (b) Relation extraction

Fig. 2. Impact of regularization parameter β

and DistMult, JointAS achieves the same performance, while JOINER further outperforms JointAS by 1.4%. Note that all KG embeddings baselines and JointAS show the same results on this task, as we report the optimal accuracy by tuning the weights for the weighted sum process, which indeed weakens the impact of individual techniques.

E. Impact of Regularization Parameter

The regularization parameter β controls to what extent we *jointly* learn embeddings from the text and KG. A smaller value of β implies that we learn less from the set of anchors that link the two data sources. In this experiment, by varying β on a log scale, we report the performance for the analogical reasoning and relation extraction tasks in Figure 2. First, we observe that neither a too small nor a too big value of β results in optimal performance. On one hand, a too small value of β leads to insufficient information sharing between the two data sources in the learning process, resulting in suboptimal performance. On the other hand, too big a value over-considers the anchors which dominate the learning process, thus leading to degraded performance. Our method JOINER provides hence the flexibility to achieve the best performance by tuning β. Second, we find the optimal β varies across different tasks, i.e., 0.001 for analogical reasoning and 0.005 for relation extraction, which can serve as a guideline for future work on joint text and KG embeddings.

F. Runtime Performance

We evaluate the efficiency of JOINER by comparing the learning time of different embedding methods. For a fair comparison, we focus on JointAS (AN+AA) and JOINER, as they share the same text and KG models, and also the same set of anchors. To give a reference, we also report the total learning time of the text+KG models (without joint learning). We report the embedding learning time[4] in Table III. Unsurprisingly, compared to Text+KG (without joint learning), both the joint embedding methods create some overhead, i.e., 163% and 87% additional learning time for JointAS and JOINER, respectively. More importantly, compared to JointAS which creates additional learning samples for joint learning, JOINER using regularization yields significantly less overhead (76% less learning time overhead).

TABLE III
RUNTIME PERFORMANCE

Method	Learning time (in hours)
Text+KG (without joint learning)	14.3
JointAS (AN+AA)	37.6
JOINER (ours)	26.8

V. Conclusions

This paper revisited text and KG joint embeddings and introduced JOINER, a novel joint text and KG embedding method providing the flexibility to control the amount of information shared between the two data sources during the joint learning process via regularization. Without additionally generated learning samples, it is also computationally efficient. Extensive experiments showed that compared to JointAS, a state-of-the-art joint text and KG embedding method generating additional learning samples from a set of anchors, JOINER yields better results (with 1.4%-4.3% improvement on various evaluation tasks) while significantly improving runtime performance (76% less overhead).

VI. Acknowledgement

This work was supported by the Swiss National Science Foundation under grant number 407540_167320.

References

[1] F. Sebastiani, "Machine learning in automated text categorization," *ACM Comput. Surv.*, vol. 34, no. 1, pp. 1–47, Mar. 2002. [Online]. Available: http://doi.acm.org/10.1145/505282.505283
[2] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning," in *ACL*. ACL, 2010, pp. 384–394.
[3] Z. Su, H. Xu, D. Zhang, and Y. Xu, "Chinese sentiment classification using a neural network toolword2vec," in *MFI*. IEEE, 2014, pp. 1–6.
[4] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *ICLR Workshop*, 2013.
[5] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013, pp. 3111–3119.
[6] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *ACM SIGMOD/PODS*. ACM, 2008, pp. 1247–1250.
[7] Google, https://www.google.com/intl/bn/insidesearch/features/search/knowledge.html, 2014.
[8] Wikidata, http://wikidata.org/, 2012.
[9] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *TKDE*, vol. 29, no. 12, pp. 2724–2743, 2017.

[4]Measured on a server with two CPUs (Intel Xeon E5-2620V4@2.10GHz) and 128G RAM using 30 threads.

[10] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *NIPS*, 2013, pp. 2787–2795.

[11] S. W.-t. Yih, M.-W. Chang, X. He, and J. Gao, "Semantic parsing via staged query graph generation: Question answering with knowledge base," in *ACL and IJCNLP*, 2015, pp. 1321–1331.

[12] J. Graupmann, R. Schenkel, and G. Weikum, "The spheresearch engine for unified ranked retrieval of heterogeneous xml and web documents," in *VLDB*. VLDB Endowment, 2005, pp. 529–540.

[13] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph and text jointly embedding," in *EMNLP*, 2014, pp. 1591–1601.

[14] B. Min, R. Grishman, L. Wan, C. Wang, and D. Gondek, "Distant supervision for relation extraction with an incomplete knowledge base," in *NAACL HLT*, 2013, pp. 777–782.

[15] H. Zhong, J. Zhang, Z. Wang, H. Wan, and Z. Chen, "Aligning knowledge and text embeddings by entity descriptions," in *EMNLP*, 2015, pp. 267–272.

[16] W. Fang, J. Zhang, D. Wang, Z. Chen, and M. Li, "Entity disambiguation by knowledge and text jointly embedding," in *CoNLL*, 2016, pp. 260–269.

[17] I. Yamada, H. Shindo, H. Takeda, and Y. Takefuji, "Joint learning of the embedding of words and entities for named entity disambiguation," in *CoNLL*, 2016, pp. 250–259.

[18] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *JMLR*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[19] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *JMLR*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[20] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014, pp. 1532–1543.

[21] P. Rosso, D. Yang, and P. Cudré-Mauroux, "Knowledge graph embeddings," in *Encyclopedia of Big Data Technologies.*, 2019.

[22] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes." in *AAAI*, vol. 14, 2014, pp. 1112–1119.

[23] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion." in *AAAI*, vol. 15, 2015, pp. 2181–2187.

[24] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *ACL and IJCNLP*, vol. 1, 2015, pp. 687–696.

[25] T. Ebisu and R. Ichise, "Toruse: Knowledge graph embedding on a lie group," in *AAAI*, 2018.

[26] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data." in *ICML*, vol. 11, 2011, pp. 809–816.

[27] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *ICLR*, 2015.

[28] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *ICML*, 2016, pp. 2071–2080.

[29] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *NIPS*, 2013, pp. 926–934.

[30] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2d knowledge graph embeddings," in *AAAI*, 2017, pp. 1811–1818.

[31] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *European Semantic Web Conference*. Springer, 2018, pp. 593–607.

[32] I. Balazevic, C. Allen, and T. M. Hospedales, "Hypernetwork knowledge graph embeddings," *arXiv preprint arXiv:1808.07018*, 2018.

[33] D. Q. Nguyen, T. Vu, T. D. Nguyen, D. Q. Nguyen, and D. Phung, "A capsule network-based embedding model for knowledge graph completion and search personalization," *arXiv preprint arXiv:1808.04122*, 2018.

[34] X. Han, Z. Liu, and M. Sun, "Neural knowledge acquisition via mutual attention between knowledge graph and text," in *AAAI*, 2018.

[35] D. Yang, P. Rosso, B. Li, and P. Cudre-Mauroux, "Nodesketch: Highly-efficient graph embeddings via recursive sketching," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19, 2019, pp. 1162–1172.

[36] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon, "Representing text for joint embedding of text and knowledge bases," in *EMNLP*, 2015, pp. 1499–1509.

[37] I. Yamada, H. Shindo, H. Takeda, and Y. Takefuji, "Learning distributed representations of texts and entities from knowledge base," *TACL*, vol. 5, pp. 397–411, 2017. [Online]. Available: https://www.transacl.org/ojs/index.php/tacl/article/view/1065

[38] M. Yu and M. Dredze, "Improving lexical embeddings with semantic knowledge," in *ACL*, vol. 2, 2014, pp. 545–550.

[39] J. Cheng, Z. Wang, J.-R. Wen, J. Yan, and Z. Chen, "Contextual text understanding in distributional semantic space," in *CIKM*. ACM, 2015, pp. 133–142.

[40] D. Yang, D. Zhang, Z. Yu, and Z. Wang, "A sentiment-enhanced personalized location recommendation system," in *Proceedings of the 24th ACM conference on hypertext and social media*. ACM, 2013, pp. 119–128.

[41] L. Chen, J. Jakubowicz, D. Yang, D. Zhang, and G. Pan, "Fine-grained urban event detection and characterization based on tensor cofactorization," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 3, pp. 380–391, 2016.

[42] M. Baumgartner, W. Zhang, B. Paudel, D. DellAglio, H. Chen, and A. Bernstein, "Aligning knowledge base and document embedding models using regularized multi-task learning," in *International Semantic Web Conference*. Springer, 2018, pp. 21–37.

[43] J. Weston, A. Bordes, O. Yakhnenko, and N. Usunier, "Connecting language and knowledge bases with embedding models for relation extraction," in *EMNLP*, 2013, pp. 1366–1371.

[44] M. Mintz, S. Bills, R. Snow, and D. Jurafsky, "Distant supervision for relation extraction without labeled data," in *ACL and AFNLP*. ACL, 2009, pp. 1003–1011.

[45] S. Riedel, L. Yao, and A. McCallum, "Modeling relations and their mentions without labeled text," in *ECML PKDD*. Springer, 2010, pp. 148–163.