

Compositional Vector Space Models for Knowledge Base Completion

Arvind Neelakantan, Benjamin Roth, Andrew McCallum

Department of Computer Science
University of Massachusetts, Amherst
Amherst, MA, 01003

{arvind,beroth,mccallum}@cs.umass.edu

Abstract

Knowledge base (KB) completion adds new facts to a KB by making inferences from existing facts, for example by inferring with high likelihood *nationality(X,Y)* from *bornIn(X,Y)*. Most previous methods infer simple one-hop relational synonyms like this, or use as evidence a multi-hop relational path treated as an atomic feature, like *bornIn(X,Z) → containedIn(Z,Y)*. This paper presents an approach that reasons about conjunctions of multi-hop relations *non-atomically*, composing the implications of a path using a recurrent neural network (RNN) that takes as inputs vector embeddings of the binary relation in the path. Not only does this allow us to generalize to paths unseen at training time, but also, with a single high-capacity RNN, to predict new relation types not seen when the compositional model was trained (zero-shot learning). We assemble a new dataset of over 52M relational triples, and show that our method improves over a traditional classifier by 11%, and a method leveraging pre-trained embeddings by 7%.

1 Introduction

Constructing large knowledge bases (KBs) supports downstream reasoning about resolved entities and their relations, rather than the noisy textual evidence surrounding their natural language mentions. For this reason KBs have been of increasing interest in both industry and academia (Bollacker et al., 2008; Suchanek et al., 2007; Carlson et al., 2010). Such KBs typically contain many millions of facts, most of them (entity1,relation,entity2) “triples” (also known as binary relations) such as (*Barack Obama*, *presi-*

dentOf, *USA*) and (*Brad Pitt*, *marriedTo*, *Angelina Jolie*).

However, even the largest KBs are woefully incomplete (Min et al., 2013), missing many important facts, and therefore damaging their usefulness in downstream tasks. Ironically, these missing facts can frequently be inferred from other facts already in the KB, thus representing a sort of inconsistency that can be repaired by the application of an automated process. The addition of new triples by leveraging existing triples is typically known as *KB completion*.

Early work on this problem focused on learning symbolic rules. For example, Schoenmackers et al. (2010) learns Horn clauses predictive of new binary relations by exhaustively exploring relational paths of increasing length, and selecting those surpassing an accuracy threshold. (A “path” is a sequence of triples in which the second entity of each triple matches the first entity of the next triple.) Lao et al. (2011) introduced the *Path Ranking Algorithm (PRA)*, which greatly improves efficiency and robustness by replacing exhaustive search with random walks, and using unique paths as features in a per-target-relation binary classifier. A typical predictive feature learned by PRA is that *CountryOfHeadquarters(X, Y)* is implied by *IsBasedIn(X,A)* and *StateLocatedIn(A, B)* and *CountryLocatedIn(B, Y)*. Given *IsBasedIn(Microsoft, Seattle)*, *StateLocatedIn(Seattle, Washington)* and *CountryLocatedIn(Washington, USA)*, we can infer the fact *CountryOfHeadquarters(Microsoft, USA)* using the predictive feature. In later work, Lao et al. (2012) greatly increase available raw material for paths by augmenting KB-schema relations with relations defined by the text connecting mentions of entities in a large corpus (also known as OpenIE relations (Banko et al., 2007)).

However, these symbolic methods can produce many millions of distinct paths, each of which is categorically distinct, treated by PRA as a dis-

what does
this mean

tinct feature. (See Figure 1.) Even putting aside the OpenIE relations, this limits the applicability of these methods to modern KBs that have thousands of relation types, since the number of distinct paths increases rapidly with the number of relation types. If textually-defined OpenIE relations are included, the problem is obviously far more severe.

Better generalization can be gained by operating on **embedded vector representations of relations**, in which vector similarity can be interpreted as **semantic similarity**. For example, Bordes et al. (2013) learn low-dimensional vector representations of entities and KB relations, such that vector differences between two entities should be close to the vectors associated with their relations. This approach can find relation synonyms, and thus perform a kind of one-to-one, non-path-based relation prediction for KB completion. Similarly Nickel et al. (2011) and Socher et al. (2013a) perform KB completion by **learning embeddings of relations, but based on matrices or tensors**. Universal schema (Riedel et al., 2013) learns to perform relation prediction cast as matrix completion (likewise using vector embeddings), but predicts textually-defined OpenIE relations as well as KB relations, and embeds entity-pairs in addition to individual entities. Like all of the above, it also reasons about individual relations, not the evidence of a connected path of relations.

This paper proposes an approach combining the advantages of (a) reasoning about conjunctions of relations connected in a path, and (b) generalization through vector embeddings, and (c) reasoning non-atomically and compositionally about the elements of the path, for further generalization.

Our method uses recurrent neural networks (RNNs) (Werbos, 1990) to compose the semantics of relations in an arbitrary-length path. At each path-step it consumes both the vector embedding of the next relation, and the vector representing the path-so-far, then outputs a composed vector (representing the extended path-so-far), which will be the input to the next step. After consuming a path, the RNN should output a vector in the semantic neighborhood of the relation between the first and last entity of the path. For example, after consuming the relation vectors along the path *Melinda Gates* \rightarrow *Bill Gates* \rightarrow *Microsoft* \rightarrow *Seattle*, our method produces a vector very close to the relation *livesIn*.

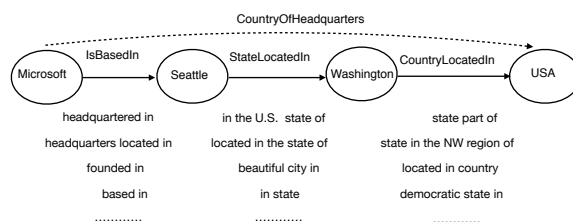


Figure 1: Semantically similar paths connecting entity pair (Microsoft, USA).

Our compositional approach allow us at test time to make predictions from paths that were unseen during training, because of the generalization provided by vector neighborhoods, and because they are composed in non-atomic fashion. This allows our model to seamlessly perform inference on many millions of paths in the KB graph. In most of our experiments, we learn a separate RNN for predicting each relation type, but alternatively, by learning a single high-capacity composition function for all relation types, our method can perform zero-shot learning—predicting new relation types for which the composition function was never explicitly trained.

Related to our work, new versions of PRA (Gardner et al., 2013; Gardner et al., 2014) use pre-trained vector representations of relations to alleviate its feature explosion problem—but the core mechanism continues to be a classifier based on atomic-path features. In the 2013 work many paths are collapsed by clustering paths according to their relations’ embeddings, and substituting cluster ids for the original relation types. In the 2014 work unseen paths are mapped to nearby paths seen at training time, where nearness is measured using the embeddings. Neither is able to perform zero-shot learning since there must be a classifier for each predicted relation type. Furthermore their pre-trained vectors do not have the opportunity to be tuned to the KB completion task because the two sub-tasks are completely disentangled.

An additional contribution of our work is a new large-scale data set of over 52 million triples, and its preprocessing for purposes of path-based KB completion (can be downloaded from <http://iesl.cs.umass.edu/downloads/inferencerules/release.tar.gz>). The dataset is build from the combination of Freebase (Bollacker et al., 2008) and Google’s entity linking in ClueWeb (Orr et al., 2013). Rather than Gardner’s 1000 distinct paths per relation type, we have over 2 million. Rather than Gardner’s 200

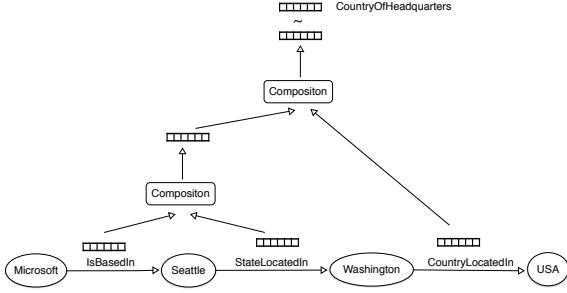


Figure 2: Vector Representations of the paths are computed by applying the composition function recursively.

entity pairs, we use over 10k. All experimental comparisons below are performed on this new data set.

On this challenging large-scale dataset our compositional method outperforms PRA (Lao et al., 2012), and Cluster PRA (Gardner et al., 2013) by 11% and 7% respectively. A further contribution of our work is a new, surprisingly strong baseline method using classifiers of path bigram features, which beats PRA and Cluster PRA, and statistically ties our compositional method. Our analysis shows that our method has substantially different strengths than the new baseline, and the combination of the two yields a 15% improvement over Gardner et al. (2013). We also show that our zero-shot model is indeed capable of predicting new unseen relation types.

2 Background

We give background on PRA which we use to obtain a set of paths connecting the entity pairs and the RNN model which we employ to model the composition function.

2.1 Path Ranking Algorithm

Since it is impractical to exhaustively obtain the set of all paths connecting an entity pair in the large KB graph, we use PRA (Lao et al., 2011) to obtain a set of paths connecting the entity pairs. Given a training set of entity pairs for a relation, PRA heuristically finds a set of paths by performing random walks from the source and target nodes keeping the most common paths. We use PRA to find millions of distinct paths per relation type. We do not use the random walk probabilities given by PRA since using it did not yield improvements in our experiments.

2.2 Recurrent Neural Networks

Recurrent neural network (RNN) (Werbos, 1990) is a neural network that constructs vector representation for sequences (of any length). For example, a RNN model can be used to construct vector representations for phrases or sentences (of any length) in natural language by applying a composition function (Mikolov et al., 2010; Sutskever et al., 2014; Vinyals et al., 2014). The vector representation of a phrase (w_1, w_2) consisting of words w_1 and w_2 is given by $f(W[v(w_1); v(w_2)])$ where $v(w) \in \mathbb{R}^d$ is the vector representation of w , f is an element-wise non linearity function, $[a; b]$ represents the concatenation two vectors a and b along with a bias term, and $W \in \mathbb{R}^{d \times 2d+1}$ is the composition matrix. This operation can be repeated to construct vector representations of longer phrases.

3 Recurrent Neural Networks for KB Completion

This paper proposes a RNN model for KB completion that reasons on the paths connecting an entity pair to predict missing relation types. The vector representations of the paths (of any length) in the KB graph are computed by applying the composition function recursively as shown in Figure 2. To compute the vector representations for the higher nodes in the tree, the composition function consumes the vector representation of the node’s two children nodes and outputs a new vector of the same dimension. Predictions about missing relation types are made by comparing the vector representation of the path with the vector representation of the relation using the sigmoid function.

We represent each binary relation using a d -dimensional real valued vector. We model composition using recurrent neural networks (Werbos, 1990). We learn a separate composition matrix for every relation that is predicted.

Let $v_r(\delta) \in \mathbb{R}^d$ be the vector representation of relation δ and $v_p(\pi) \in \mathbb{R}^d$ be the vector representation of path π . $v_p(\pi)$ denotes the relation vector if path π is of length one. To predict relation $\delta = \text{CountryOfHeadquarters}$, the vector representation of the path $\pi = \text{IsBasedIn} \rightarrow \text{StateLocatedIn}$ containing two relations *IsBasedIn* and *StateLocatedIn* is computed by (Figure 2),

$$v_p(\pi) = f(W_\delta[v_r(\text{IsBasedIn}); v_r(\text{StateLocatedIn})])$$

where $f = \text{sigmoid}$ is the element-wise non-linearity function, $W_\delta \in \mathbb{R}^{d \times 2d+1}$ is the composition matrix for $\delta = \text{CountryOfHeadquarters}$ and $[a; b]$ represents the concatenation of two vectors $a \in \mathbb{R}^d, b \in \mathbb{R}^d$ along with a bias feature to get a new vector $[a; b] \in \mathbb{R}^{2d+1}$.

The vector representation of the path $\Pi = \text{IsBasedIn} \rightarrow \text{StateLocatedIn} \rightarrow \text{CountryLocatedIn}$ in Figure 2 is computed similarly by,

$$v_p(\Pi) = f(W_\delta[v_p(\pi); v_r(\text{CountryLocatedIn})])$$

where $v_p(\pi)$ is the vector representation of path $\text{IsBasedIn} \rightarrow \text{StateLocatedIn}$. While computing the vector representation of a path we always traverse left to right, composing the relation vector in the right with the accumulated path vector in the left¹. This makes our model a recurrent neural network (Werbos, 1990).

Finally, we make a prediction regarding *CountryOfHeadquarters(Microsoft, USA)* using the path $\Pi = \text{IsBasedIn} \rightarrow \text{StateLocatedIn} \rightarrow \text{CountryLocatedIn}$ by comparing the vector representation of the path ($v_p(\Pi)$) with the vector representation of the relation *CountryOfHeadquarters* ($v_r(\text{CountryOfHeadquarters})$) using the sigmoid function.

3.1 Model Training

We train the model with the existing facts in a KB using them as positive examples and negative examples are obtained by treating the unobserved instances as negative examples (Mintz et al., 2009; Lao et al., 2011; Riedel et al., 2013; Bordes et al., 2013). Unlike in previous work that use RNNs (Socher et al., 2011; Iyyer et al., 2014; Irsoy and Cardie, 2014), a challenge with using them for our task is that among the set of paths connecting an entity pair, we do not observe which of the path(s) is predictive of a relation. We select the path that is closest to the relation type to be predicted in the vector space. This not only allows for faster training (compared to marginalization) but also gives improved performance. This technique has been successfully used in models other than RNNs previously (Weston et al., 2013; Nee-lakantan et al., 2014).

¹we did not get significant improvements when we tried more sophisticated ordering schemes for computing the path representations.

Algorithm 1 Training Algorithm of RNN model for relation δ

```

1: Input:  $\Lambda_\delta = \Lambda_\delta^+ \cup \Lambda_\delta^-$ ,  $\Phi_\delta$ , number of iterations  $T$ , mini-batch size  $B$ 
2: Initialize  $v_r, W_\delta$  randomly
3: for  $t = 1, 2, \dots, T$  do
4:    $\nabla v_r = 0, \nabla W_\delta = 0$  and  $b = 0$ 
5:   for  $\lambda = (\gamma, \delta) \in \Lambda_\delta$  do
6:      $\mu_\lambda = \arg \max_{\pi \in \Phi_\delta(\gamma)} v_p(\pi) \cdot v_r(\delta)$ 
7:     Accumulate gradients to  $\nabla v_r, \nabla W_\delta$ 
8:     using path  $\mu_\lambda$ .
9:      $b = b + 1$ 
10:    if  $b = B$  then
11:      Gradient Update for  $v_r, W_\delta$ 
12:       $\nabla v_r = 0, \nabla W_\delta = 0$  and  $b = 0$ 
13:    end if
14:  end for
15:  if  $b > 0$  then
16:    Gradient Update for  $v_r, W_\delta$ 
17:  end if
18: end for
19: Output:  $v_r, W_\delta$ 

```

We assume that we are given a KB (for example, Freebase enriched with SVO triples) containing a set of entity pairs Γ , set of relations Δ and a set of observed facts Λ^+ where $\forall \lambda = (\gamma, \delta) \in \Lambda^+ (\gamma \in \Gamma, \delta \in \Delta)$ indicates a positive fact that entity pair γ is in relation δ . Let $\Phi_\delta(\gamma)$ denote the set of paths connecting entity pair γ given by PRA for predicting relation δ .

In our task, we only observe the set of paths connecting an entity pair but we do not observe which of the path(s) is predictive of the fact. We treat this as a latent variable (μ_λ for the fact λ) and we assign μ_λ the path whose vector representation has maximum dot product with the vector representation of the relation to be predicted. For example, μ_λ for the fact $\lambda = (\gamma, \delta) \in \Lambda^+$ is given by,

$$\mu_\lambda = \arg \max_{\pi \in \Phi_\delta(\gamma)} v_p(\pi) \cdot v_r(\delta)$$

During training, we assign μ_λ using the current parameter estimates. We use the same procedure to assign μ_λ for unobserved facts that are used as negative examples during training.

We train a separate RNN model for predicting each relation and the parameters of the model for predicting relation $\delta \in \Delta$ are $\Theta = \{v_r(\omega) \mid \omega \in \Delta, W_\delta\}$. Given a training set consisting of posi-

tive (Λ_δ^+) and negative (Λ_δ^-) instances² for relation δ , the parameters are trained to maximize the log likelihood of the training set with L-2 regularization.

$$\Theta^* = \arg \max_{\Theta} \sum_{\lambda=(\gamma,\delta) \in \Lambda_\delta^+} P(y_\lambda = 1; \Theta) + \sum_{\lambda=(\gamma,\delta) \in \Lambda_\delta^-} P(y_\lambda = 0; \Theta) - \rho \|\Theta\|^2$$

where y_λ is a binary random variable which takes the value 1 if the fact λ is true and 0 otherwise, and the probability of a fact $P(y_\lambda = 1; \Theta)$ is given by,

$$P(y_\lambda = 1; \Theta) = \text{sigmoid}(v_p(\mu_\lambda) \cdot v_r(\delta))$$

where $\mu_\lambda = \arg \max_{\pi \in \Phi_\delta(\gamma)} v_p(\pi) \cdot v_r(\delta)$

and $P(y_\lambda = 0; \Theta) = 1 - P(y_\lambda = 1; \Theta)$. The relation vectors and the composition matrix are initialized randomly. We train the network using backpropagation through structure (Goller and Küchler, 1996).

4 Zero-shot KB Completion

The KB completion task involves predicting facts on thousands of relations types and it is highly desirable that a method can infer facts about relation types without directly training for them. Given the vector representation of the relations, we show that our model described in the previous section is capable of predicting relational facts without explicitly training for the target (or test) relation types (zero-shot learning).

In zero-shot or zero-data learning (Larochelle et al., 2008; Palatucci et al., 2009), some labels or classes are not available during training the model and only a description of those classes are given at prediction time. We make two modifications to the model described in the previous section, (1) learn a general composition matrix, and (2) fix relation vectors with pre-trained vectors, so that we can predict relations that are unseen during training. This ability of the model to generalize to unseen relations is beyond the capabilities of all previous methods for KB inference (Schoenmackers et al., 2010; Lao et al., 2011; Gardner et al., 2013; Gardner et al., 2014).

We learn a general composition matrix for all relations instead of learning a separate composition matrix for every relation to be predicted. So,

²we sub-sample a portion of the set of all unobserved instances.

for example, the vector representation of the path $\pi = \text{IsBasedIn} \rightarrow \text{StateLocatedIn}$ containing two relations *IsBasedIn* and *StateLocatedIn* is computed by (Figure 2),

$$v_p(\pi) = f(W[v_r(\text{IsBasedIn}); v_r(\text{StateLocatedIn})])$$

where $W \in \mathbb{R}^{d \times 2d+1}$ is the general composition matrix.

We initialize the vector representations of the binary relations (v_r) using the representations learned in Riedel et al. (2013) and do not update them during training. The relation vectors are not updated because at prediction time we would be predicting relation types which are never seen during training and hence their vectors would never get updated. We learn only the general composition matrix in this model. We train a single model for a set of relation types by replacing the sigmoid function with a softmax function while computing probabilities and the parameters of the composition matrix are learned using the available training data containing instances of few relations. The other aspects of the model remain unchanged.

To predict facts whose relation types are unseen during training, we compute the vector representation of the path using the general composition matrix and compute the probability of the fact using the pre-trained relation vector. For example, using the vector representation of the path $\Pi = \text{IsBasedIn} \rightarrow \text{StateLocatedIn} \rightarrow \text{CountryLocatedIn}$ in Figure 2, we can predict any relation irrespective of whether they are seen at training by comparing it with the pre-trained relation vectors.

5 Experiments

The hyperparameters of all the models were tuned on the same held-out development data. All the neural network models are trained for 150 iterations using 50 dimensional relation vectors, and we set the L2-regularizer and learning rate to 0.0001 and 0.1 respectively. We halved the learning rate after every 60 iterations and use mini-batches of size 20. The neural networks and the classifiers were optimized using AdaGrad (Duchi et al., 2011).

5.1 Data

We ran experiments on Freebase (Bollacker et al., 2008) enriched with information from ClueWeb.

Entities	18M
Freebase triples	40M
ClueWeb triples	12M
Relations	25,994
Relation types tested	46
Avg. paths/relation	2.3M
Avg. training facts/relation	6638
Avg. positive test instances/relation	3492
Avg. negative test instances/relation	43,160

Table 1: Statistics of our dataset.

We use the publicly available entity links to Freebase in the ClueWeb dataset (Orr et al., 2013). Hence, we create nodes only for Freebase entities in our KB graph. We remove facts containing /type/object/type as they do not give useful predictive information for our task. We get triples from ClueWeb by considering sentences that contain two entities linked to Freebase. We extract the phrase between the two entities and treat them as the relation types. For phrases that are of length greater than four we keep only the first and last two words. This helps us to avoid the time consuming step of dependency parsing the sentence to get the relation type. These triples are similar to facts obtained by OpenIE (Banko et al., 2007). To reduce noise, we select relation types that occur at least 50 times. We evaluate on 46 relation types in Freebase that have the most number of instances. The methods are evaluated on a subset of facts in Freebase that were hidden during training. Table 1 shows important statistics of our dataset.

5.2 Predictive Paths

Table 2 shows predictive paths for 4 relations learned by the RNN model. The high quality of unseen paths is indicative of the fact that the RNN model is able to generalize to paths that are never seen during training.

5.3 Results

Using our dataset, we compare the performance of the following methods:

PRA Classifier is the method in Lao et al. (2012) which trains a logistic regression classifier by creating a feature for every path type.

Cluster PRA Classifier is the method in Gardner et al. (2013) which replaces relation types from ClueWeb triples with their cluster membership in the KB graph before the path finding step. Af-

ter this step, their method proceeds in exactly the same manner as Lao et al. (2012) training a logistic regression classifier by creating a feature for every path type. We use pre-trained relation vectors from Riedel et al. (2013) and use k-means clustering to cluster the relation types to 25 clusters as done in Gardner et al. (2013).

Composition-Add uses a simple element-wise addition followed by sigmoid non-linearity as the composition function similar to Yang et al. (2014). **RNN-random** is the supervised RNN model described in section 3 with the relation vectors initialized randomly.

RNN is the supervised RNN model described in section 3 with the relation vectors initialized using the method in Riedel et al. (2013).

PRA Classifier-b is our simple extension to the method in Lao et al. (2012) which additionally uses bigrams in the path as features. We add a special *start* and *stop* symbol to the path before computing the bigram features.

Cluster PRA Classifier-b is our simple extension to the method in Gardner et al. (2013) which additionally uses bigram features computed as previously described.

RNN + PRA Classifier combines the predictions of RNN and PRA Classifier. We combine the predictions by assigning the score of a fact as the sum of their rank in the two models after sorting them in ascending order.

RNN + PRA Classifier-b combines the predictions of RNN and PRA Classifier-b using the technique described previously.

Table 3 shows the results of our experiments. The method described in Gardner et al. (2014) is not included in the table since the publicly available implementation does not scale to our large dataset. First, we show that it is better to train the models using all the path types instead of using only the top 1,000 path types as done in previous work (Gardner et al., 2013; Gardner et al., 2014). We can see that the RNN model performs significantly better than the baseline methods of Lao et al. (2012) and Gardner et al. (2013). The performance of the RNN model is not affected by initialization since using random vectors and pre-trained vectors results in similar performance.

A surprising result is the impressive performance of our simple extension to the classifier approach. After the addition of bigram features, the naive PRA method is as effective as the Clus-

Relation: /book/written_work/original_language/ (book “x” written in language “y”) Seen paths: /book/written_work/previous_in_series → /book/written_work/author → /people/person/nationality → /people/person/nationality ⁻¹ → /people/person/languages /book/written_work/author → /people/ethnicity/people ⁻¹ → /people/ethnicity/languages_spoken Unseen paths: “in” ⁻¹ - “writer” ⁻¹ → /people/person/nationality ⁻¹ → /people/person/languages /book/written_work/author → addresses → /people/person/nationality ⁻¹ → /people/person/languages
Relation: /people/person/place_of_birth/ (person “x” born in place “y”) Seen paths: “was,born,in” → /location/mailling_address/citytown ⁻¹ → /location/mailling_address/state_province_region “from” → /location/location/contains ⁻¹ Unseen paths: “born,in” → /location/location/contains → “near” ⁻¹ “was,born,in” → commonly_known,as ⁻¹
Relation: /geography/river/cities/ (river “x” flows through or borders “y”) Seen paths: “at” → /location/location/contains ⁻¹ “meets,the” → /transportation/bridge/body_of_water_spanned ⁻¹ → /location/location/contains ⁻¹ → “in” Unseen paths: /geography/lake/outflow ⁻¹ → /location/location/contains ⁻¹ /geography/lake/outflow ⁻¹ → /location/location/contains ⁻¹ → “near”
Relation: /people/family/members/ (person “y” part of family “x”) Seen paths: /royalty/monarch/royal_line ⁻¹ → /people/person/children → /royalty/monarch/royal_line → /royalty/royal_line/monarchs_from_this_line /royalty/royal_line/monarchs_from_this_line → /people/person/parents ⁻¹ → /people/person/parents ⁻¹ → /people/person/parents ⁻¹ Unseen paths: /royalty/monarch/royal_line ⁻¹ → “leader” ⁻¹ → “king” → “was,married,to” ⁻¹ “of,the” ⁻¹ → “but,also,of” → “married” → “defended” ⁻¹

Table 2: Predictive paths, according to the *RNN* model, for 4 target relations. Two examples of seen and unseen paths are shown for each target relation. Inverse relations are marked by ⁻¹, i.e. $r(x, y) \implies r^{-1}(y, x)$, $\forall (x, y) \in r$. Relations within quotes are OpenIE (textual) relation types.

	train with top 1000 paths	train with all paths
Method	MAP	MAP
<i>PRA Classifier</i>	43.46	51.31
<i>Cluster PRA Classifier</i>	46.26	53.23
<i>Composition-Add</i>	40.23	45.37
<i>RNN-random</i>	45.52	56.91
<i>RNN</i>	46.61	56.95
<i>PRA Classifier-b</i>	48.09	58.13
<i>Cluster PRA Classifier-b</i>	48.72	58.02
<i>RNN + PRA Classifier</i>	49.92	58.42
<i>RNN + PRA Classifier-b</i>	51.94	61.17

Table 3: Results comparing different methods on 46 types. All the methods perform better when trained using all the paths than training using the top 1,000 paths. When training with all the paths, *RNN* performs significantly ($p < 0.005$) better than *PRA Classifier* and *Cluster PRA Classifier*. The small difference in performance between *RNN* and both *PRA Classifier-b* and *Cluster PRA Classifier-b* is not statistically significant. The best results are obtained by combining the predictions of *RNN* with *PRA Classifier-b* which performs significantly ($p < 10^{-5}$) better than both *PRA Classifier-b* and *Cluster PRA Classifier-b*.

ter PRA method. The small difference in performance between *RNN* and both *PRA Classifier-b* and *Cluster PRA Classifier-b* is not statistically significant. We conjecture that our method has

substantially different strengths than the new baseline. While the classifier with bigram features has an ability to accurately memorize important local structure, the *RNN* model generalizes better to un-

	train with top 1000 paths	train with all paths
Method	MAP	MAP
RNN	43.82	50.10
zero-shot	19.28	20.61
Random	7.59	

Table 4: Results comparing the zero-shot model with supervised RNN and a random baseline on 10 types. RNN is the fully supervised model described in section 3 while zero-shot is the model described in section 4. The zero-shot model without explicitly training for the target relation types achieves impressive results by performing significantly ($p < 0.05$) better than a random baseline.

seen paths that are very different from the paths seen in training. Empirically, combining the predictions of *RNN* and *PRA Classifier-b* achieves a statistically significant gain over *PRA Classifier-b*.

5.3.1 Zero-shot

Table 4 shows the results of the zero-shot model described in section 4 compared with the fully supervised RNN model (section 3) and a baseline that produces a random ordering of the test facts. We evaluate on randomly selected 10 (out of 46) relation types, hence for the fully supervised version we train 10 RNNs, one for each relation type. For evaluating the zero-shot model, we randomly split the relations into two sets of equal size and train a zero-shot model on one set and test on the other set. So, in this case we have two RNNs making predictions on relation types that they have never seen during training. As expected, the fully supervised RNN outperforms the zero-shot model by a large margin but the zero-shot model without using any direct supervision clearly performs much better than a random baseline.

5.3.2 Discussion

To investigate whether the performance of the RNNs were affected by multiple local optima issues, we combined the predictions of five different RNNs trained using all the paths. Apart from *RNN* and *RNN-random*, we trained three more RNNs with different random initialization and the performance of the three RNNs individually are 57.09, 57.11 and 56.91. The performance of the ensemble is 59.16 and their performance stopped improving after using three RNNs. So, this indicates that even though multiple local optima affects the

performance, it is likely not the only issue since the performance of the ensemble is still less than the performance of *RNN + PRA Classifier-b*.

We suspect the RNN model does not capture some of the important local structure as well as the classifier using bigram features. To overcome this drawback, in future work, we plan to explore compositional models that have a longer memory (Hochreiter and Schmidhuber, 1997; Cho et al., 2014; Mikolov et al., 2014). We also plan to include vector representations for the entities and develop models that address the issue of polysemy in verb phrases (Cheng et al., 2014).

6 Related Work

KB Completion includes methods such as Lin and Pantel (2001), Yates and Etzioni (2007) and Berant et al. (2011) that learn inference rules of length one. Schoenmackers et al. (2010) learn general inference rules by considering the set of all paths in the KB and selecting paths that satisfy a certain precision threshold. Their method does not scale well to modern KBs and also depends on carefully tuned thresholds. Lao et al. (2011) train a simple logistic regression classifier with NELL KB paths as features to perform KB completion while Gardner et al. (2013) and Gardner et al. (2014) extend it by using pre-trained relation vectors to overcome feature sparsity. Recently, Yang et al. (2014) learn inference rules using simple element-wise addition or multiplication as the composition function.

Compositional Vector Space Models have been developed to represent phrases and sentences in natural language as vectors (Mitchell and Lapata, 2008; Baroni and Zamparelli, 2010; Yessenalina and Cardie, 2011). Neural networks have been successfully used to learn vector representations of phrases using the vector representations of the words in that phrase. Recurrent neural networks have been used for many tasks such as language modeling (Mikolov et al., 2010), machine translation (Sutskever et al., 2014) and parsing (Vinyals et al., 2014). Recursive neural networks, a more general version of the recurrent neural networks have been used for many tasks like parsing (Socher et al., 2011), sentiment classification (Socher et al., 2012; Socher et al., 2013c; Irsoy and Cardie, 2014), question answering (Iyyer et al., 2014) and natural language logical semantics (Bowman et al., 2014). Our overall approach is

similar to RNNs with attention (Bahdanau et al., 2014; Graves, 2013) since we select a path among the set of paths connecting the entity pair to make the final prediction.

Zero-shot or zero-data learning was introduced in Larochelle et al. (2008) for character recognition and drug discovery. Palatucci et al. (2009) perform zero-shot learning for neural decoding while there has been plenty of work in this direction for image recognition (Socher et al., 2013b; Frome et al., 2013; Norouzi et al., 2014).

7 Conclusion

We develop a compositional vector space model for knowledge base completion using recurrent neural networks. In our challenging large-scale dataset available at <http://iesl.cs.umass.edu/downloads/inferencerules/release.tar.gz>, our method outperforms two baseline methods and performs competitively with a modified stronger baseline. The best results are obtained by combining the predictions of our model with the predictions of the modified baseline which achieves a 15% improvement over Gardner et al. (2013). We also show that our model has the ability to perform zero-shot inference.

Acknowledgments

We thank Matt Gardner for releasing the PRA code, and for answering numerous question about the code and data. We also thanks the Stanford NLP group for releasing the neural networks code. This work was supported in part by the Center for Intelligent Information Retrieval, in part by DARPA under agreement number FA8750-13-2-0020, in part by an award from Google, and in part by NSF grant #CNS-0958392. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *ArXiv*.

Michele Banko, Michael J Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *International Joint Conference on Artificial Intelligence*.

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Empirical Methods in Natural Language Processing*.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *Association for Computational Linguistics*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*.

Samuel R. Bowman, Christopher Potts, and Christopher D Manning. 2014. Recursive neural networks for learning logical semantics. In *CoRR*.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, and A. 2010. Toward an architecture for never-ending language learning. In *In AAAI*.

Cheng, Jianpeng Kartsaklis, and Edward Grefenstette. 2014. Investigating the role of prior disambiguation in deep-learning compositional models of meaning. In *In Learning Semantics workshop NIPS*.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In *Workshop on Syntax, Semantics and Structure in Statistical Translation*.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. In *Journal of Machine Learning Research*.

Andrea Frome, Gregory S. Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc’Aurelio Ranzato, and Tomas Mikolov. 2013. Devise: A deep visual-semantic embedding model. In *Neural Information Processing Systems*.

Matt Gardner, Partha Pratim Talukdar, Bryan Kisiel, and Tom M. Mitchell. 2013. Improving learning and inference in a large knowledge-base using latent syntactic cues. In *Empirical Methods in Natural Language Processing*.

- Matt Gardner, Partha Talukdar, Jayant Krishnamurthy, and Tom Mitchell. 2014. Incorporating vector space similarity in random walk inference over knowledge bases. In *Empirical Methods in Natural Language Processing*.
- Christoph Goller and Andreas Küchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *IEEE Transactions on Neural Networks*.
- Alex Graves. 2013. Generating sequences with recurrent neural networks. In *ArXiv*.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. In *Neural Computation*.
- Ozan Irsoy and Claire Cardie. 2014. Deep recursive neural networks for compositionality in language. In *Neural Information Processing Systems*.
- Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. 2014. A neural network for factoid question answering over paragraphs. In *Empirical Methods in Natural Language Processing*.
- Ni Lao, Tom Mitchell, and William W. Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *Conference on Empirical Methods in Natural Language Processing*.
- Ni Lao, Amarnag Subramanya, Fernando Pereira, and William W. Cohen. 2012. Reading the web with learned syntactic-semantic inference rules. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.
- Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. 2008. Zero-data learning of new tasks. In *National Conference on Artificial Intelligence*.
- Dekang Lin and Patrick Pantel. 2001. Dirt - discovery of inference rules from text. In *International Conference on Knowledge Discovery and Data Mining*.
- Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Annual Conference of the International Speech Communication Association*.
- Tomas Mikolov, Armand Joulin, Sumit Chopra, Michaël Mathieu, and Marc’Aurelio Ranzato. 2014. Learning longer memory in recurrent neural networks. In *CoRR*.
- Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. Distant supervision for relation extraction with an incomplete knowledge base. In *HLT-NAACL*, pages 777–782.
- Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Association for Computational Linguistics and International Joint Conference on Natural Language Processing*.
- Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Association for Computational Linguistics*.
- Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Empirical Methods in Natural Language Processing*.
- Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegl. 2011. A three-way model for collective learning on multi-relational data. In *International Conference on Machine Learning*.
- Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg Corrado, and Jeffrey Dean. 2014. Zero-shot learning by convex combination of semantic embeddings. In *International Conference on Learning Representations*.
- Dave Orr, Amarnag Subramanya, Evgeniy Gabilovich, and Michael Ringgaard. 2013. 11 billion clues in 800 million documents: A web research corpus annotated with freebase concepts. <http://googleresearch.blogspot.com/2013/07/11-billion-clues-in-800-million.html>.
- Mark Palatucci, Dean Pomerleau, Geoffrey Hinton, and Tom Mitchell. 2009. Zero-shot learning with semantic output codes. In *Neural Information Processing Systems*.
- Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *HLT-NAACL*.
- Stefan Schoenmackers, Oren Etzioni, Daniel S. Weld, and Jesse Davis. 2010. Learning first-order horn clauses from web text. In *Empirical Methods in Natural Language Processing*.
- Richard Socher, Cliff Chiung-Yu Lin, Christopher D. Manning, and Andrew Y. Ng. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*.
- Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.
- Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013a. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*.
- Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. 2013b. Zero-shot learning through cross-modal transfer. In *Neural Information Processing Systems*.

- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013c. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing*.
- Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*.
- Ilya Sutskever, Oriol Vinyals, and Quoc V. V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*.
- Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2014. Grammar as a foreign language. In *CoRR*.
- Paul Werbos. 1990. Backpropagation through time: what it does and how to do it. In *IEEE*.
- Jason Weston, Ron Weiss, and Hector Yee. 2013. Nonlinear latent factorization by embedding multiple user interests. In *ACM International Conference on Recommender Systems*.
- Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. In *CoRR*.
- Alexander Yates and Oren Etzioni. 2007. Unsupervised resolution of objects and relations on the web. In *North American Chapter of the Association for Computational Linguistics*.
- Ainur Yessenalina and Claire Cardie. 2011. Compositional matrix-space models for sentiment analysis. In *Empirical Methods in Natural Language Processing*.