Adaptive Convolution for Multi-Relational Learning

Xiaotian Jiang[1], **Quan Wang**[2], **Bin Wang**[3]
[1]Cloud and Smart Industries Group, Tencent
[2]Baidu Inc.
[3]Xiaomi AI Lab
johnxtjiang@tencent.com, quanwang1012@gmail.com,
wangbin11@xiaomi.com

Abstract

We consider the problem of learning distributed representations for entities and relations of multi-relational data so as to predict missing links therein. Convolutional neural networks have recently shown their superiority for this problem, bringing increased model expressiveness while remaining parameter efficient. Despite the success, previous convolution designs fail to model full interactions between input entities and relations, which potentially limits the performance of link prediction. In this work we introduce ConvR, an adaptive convolutional network designed to maximize entity-relation interactions in a convolutional fashion. ConvR adaptively constructs convolution filters from relation representations, and applies these filters across entity representations to generate convolutional features. As such, ConvR enables rich interactions between entity and relation representations at diverse regions, and all the convolutional features generated will be able to capture such interactions. We evaluate ConvR on multiple benchmark datasets. Experimental results show that: (1) ConvR performs substantially better than competitive baselines in almost all the metrics and on all the datasets; (2) Compared with state-of-the-art convolutional models, ConvR is not only more effective but also more efficient. It offers a 7% increase in MRR and a 6% increase in Hits@10, while saving 12% in parameter storage.

1 Introduction

Multi-relational data refers to directed graphs whose nodes correspond to entities and edges different types of relations between entities. An edge of the form (*subject*, *relation*, *object*) indicates that there exists a specific relation between the subject and object entities. Learning with multi-relational data plays a pivotal role in many application domains, ranging from social networks or recommender systems to large-scale knowledge bases (KBs) (Bordes et al., 2013; Jenatton et al., 2012). This work focuses on modeling multi-relational data from KBs, with the aim of predicting missing facts on KBs, a challenging task known as link prediction in statistical relational learning (SRL) (Getoor and Taskar, 2007).

Various SRL techniques (Nickel et al., 2016a) have been proposed for this task, among which vector space embedding models (Wang et al., 2017) are gaining increasing attention due to their superior performance and potential scalability. The key idea there is to learn and operate on latent features (embeddings) of entities and relations, so as to uncover non-trivial connectivity patterns in multi-relational data. Previous works of this kind tend to adopt shallow, simple models to extract latent features, e.g., the translation based models (Bordes et al., 2013; Wang et al., 2014) or the bilinear models and their variants (Jenatton et al., 2012; Yang et al., 2015; Trouillon et al., 2016). Using these simple models allows one to easily handle large-scale KBs, but usually at the cost of learning less expressive features. In fact, such simple models typically generate a single feature with each entry of the embeddings. The only way to increase the number of features (and thus their expressiveness) is to increase the embedding size (Dettmers et al., 2018). This potentially limits the performance of link prediction with a given number of parameters. To increase model expressiveness, there emerge some deeper, more complicated designs, in particular those on the basis of neural network architectures (Socher et al., 2013; Bordes et al., 2014; Dong et al., 2014; Schlichtkrull et al., 2017a). Such approaches, however, often have more parameters and are prone to overfit, at least on the (relatively small) benchmark datasets used by the scientific community (Nickel et al., 2016a).

Recently, Dettmers et al. (2018) devised ConvE,

Proceedings of NAACL-HLT 2019, pages 978–987
Minneapolis, Minnesota, June 2 - June 7, 2019.

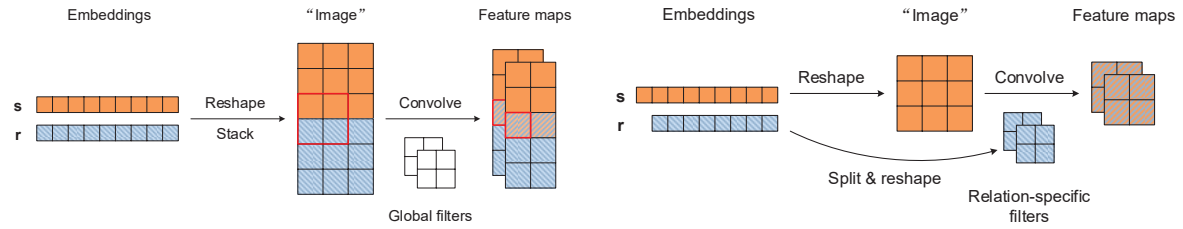
a multi-layer convolutional network which enables expressive feature learning while remaining highly parameter efficient. Given a subject-relation-object triple (s, r, o), ConvE first reshapes the vector representation of the subject s and that of the relation r into 2D matrices, and then concatenates the two matrices and feeds them into a 2D convolutional layer to extract higher-level, non-linear features, as illustrated in Figure 1(a). The resultant convolutional features are finally projected and matched with the vector representation of the object o via an inner product. Note that by sliding across the embeddings using small-sized filters, the convolution operator can easily generate much more features without increasing the embedding size. As such, ConvE offers increased expressiveness and achieves competitive performance in link prediction.

Nevertheless, despite its success, ConvE is still insufficient to fully capture the interactions between input entities and relations, which has long been recognized as crucial for modeling multi-relational data (Nickel et al., 2011; García-Durán et al., 2014; Trouillon et al., 2016). In ConvE, the (reshaped) representations of input entities and relations are simply stacked together and fed into a convolutional layer. Although 2D convolution is better than 1D convolution in modeling entity-relation interactions, typical 2D convolution with global filters on such a stacked matrix, however, can only model interactions around the concatenation line (Dettmers et al., 2018). Consider the example in Figure 1(a), where two matrices of size 3×3 are formed after reshaping, stacked, and fed as input to a convolutional layer. Convolving across the input with a global filter of size 2×2 will then be able to model interactions only in the regions where the two matrices adjoin (e.g., the region outlined in red). That means, only a small proportion of the output convolutional features (20% in this example, striped with orange and blue) will effectively capture entity-relation interactions, and the vast majority others will be entity- or relation-independent. This poses potential negative impacts on the link prediction task.

This paper, aiming at maximizing the interactions between input entities and relations, introduces ConvR, an adaptive convolutional network specifically designed for multi-relational data. As illustrated in Figure 1(b), the key idea of ConvR is to facilitate convolution across entity representations with its filters adaptively constructed from relation representations. Such adaptive convolution will model the interactions between the two types of input not only more naturally but also more effectively. Specifically, given a triple, the vector representation of the subject is reshaped and fed as input to a convolutional layer, while that of the relation is split and reshaped into a set of filters. ConvR then convolves across the input with these filters, enabling each filter (a part of the relation representation) to interact with diverse regions of the input (the entity representation). Through this adaptive convolution process, all the features generated will be able to capture entity-relation interactions (striped with orange and blue in Figure 1(b)). These convolutional features are finally projected and matched with the representation of the object.

Besides being more effective, adaptive convolution enables potentially more efficient modeling (in terms of the number of parameters). Compared with ConvE (Figure 1(a)), ConvR (Figure 1(b)) needs no global filters and generates smaller feature maps, making the follow-up projection layer roughly half as large as that of ConvE. The idea of adaptive convolution, in fact, is rather generic for the multi-relational scenario. By splitting and reshaping relation vectors, ConvR can be easily generalized to other paradigms such as 1D or 3D convolution, not restricted to the 2D setting. To facilitate a direct and fair comparison to ConvE where only 2D convolution is considered and tested, this paper takes the 2D setting as an example, and shows the superiority of ConvR over ConvE in this setting. We will investigate higher dimensional convolution in our future work.

Our contributions are as follows. (1) We propose a novel adaptive convolution model for learning with multi-relational data. Our approach, ConvR, takes full advantage of entity-relation interactions in a convolutional fashion, while still remaining highly parameter efficient. (2) We evaluate ConvR in the link prediction task on KBs and achieve very promising results on multiple benchmark datasets, including not only the popular WN18 and FB15K (Bordes et al., 2013), but also the more difficult WN18RR (Dettmers et al., 2018) and FB15K-237 (Toutanova and Chen, 2015). (3) We systematically compare the efficiency and effectiveness of ConvR and ConvE on FB15K-237, showing that ConvR can perform substantially better with a good variety of parameter settings. In particular, it offers a 7%



(a) ConvE: Convolution with global filters.

(b) ConvR: Convolution with relation-specfic filters.

Figure 1: Reshaping and convolution in ConvE and ConvR. Entity-related neurons are marked in orange, relation-related ones striped with blue backslash, and those capturing entity-relation interactions striped with slash. White blocks stand for global filters applied to all input entities and relations.

increase in MRR and a 6% increase in Hits@10, with the total parameter number only 88% as large as that of ConvE.

2 Background

We consider multi-relational data represented as a graph, which can also be formalized as a set of subject-relation-object triples $\mathcal{G} = \{(s, r, o)\} \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$. Here, $\mathcal{E}$ is the set of entities, and $\mathcal{R}$ the set of relations. Each triple (s, r, o) is composed of a subject entity $s \in \mathcal{E}$, a relation $r \in \mathcal{R}$, and an object entity $o \in \mathcal{E}$, indicating that there exists a relation of type r between the two entities s and o. Such triples are also called facts in knowledge bases (KBs).

We follow (Dettmers et al., 2018) and formalize link prediction on multi-relational data as a pointwise learning to rank problem, where the objective is to learn a scoring function $\psi : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \to \mathbb{R}$. For any input triple (s, r, o), the higher the score $\psi(s, r, o)$, the more likely the triple is true. Various statistical relational learning (SRL) techniques have been proposed for this task. See (Nickel et al., 2016a) for a thorough review of such techniques, with their application on large-scale KBs.

This paper focuses on vector space embedding models, a branch of SRL with superior performance and potential scalability. Given an input triple (s, r, o), a model of this kind first maps the entities s, o and relation r to their distributed representations (i.e., embeddings), usually vectors $\mathbf{s}, \mathbf{r}, \mathbf{o} \in \mathbb{R}^d$ for efficient learning. A score is then defined for the triple by operating on these distributed representations, i.e., $\psi(s, r, o) = \phi(\mathbf{s}, \mathbf{r}, \mathbf{o})$. A great many approaches of this kind have been devised in the last few years, where a key difference is the designing of the scoring function $\phi(\mathbf{s}, \mathbf{r}, \mathbf{o})$. See (Wang et al., 2017) for a recent survey.

3 Adaptive Convolution on Multi-relational Data

This section presents ConvR, an adaptive convolutional network specifically designed for learning with multi-relational data. The key idea of ConvR is to facilitate convolution across entity representations with its filters adaptively constructed from relation representations, so as to maximize the interactions between the two types of input. Figure 1(b) provides a simple illustration of this idea. In the rest of this section, we detail the ConvR model, discuss parameter learning of it, and show its advantages over ConvE, a convolutional network achieving promising results in multi-relational link prediction (Dettmers et al., 2018). To facilitate a direct and fair comparison to ConvE, we focus on the 2D setting. But the idea can be easily generalized to other convolution paradigms.

The ConvR model Given a triple (s, r, o), ConvR maps the two entities s, o to vectors $\mathbf{s}, \mathbf{o} \in \mathbb{R}^{d_e}$, and the relation r to vector $\mathbf{r} \in \mathbb{R}^{d_r}$, where d_e and d_r are the embedding size of entities and relations, respectively. Then, the subject vector $\mathbf{s}$ is reshaped into a 2D matrix $\mathbf{S} \in \mathbb{R}^{d_e^h \times d_e^w}$ (where $d_e = d_e^h d_e^w$) and fed as input to a convolutional layer. As shown in (Dettmers et al., 2018), using 2D rather than 1D convolution would be able to extract more feature interactions and increase model expressiveness. The relation vector $\mathbf{r}$ is further split into blocks $\mathbf{r}^{(1)}, \cdots, \mathbf{r}^{(c)}$ with equal size, where each $\mathbf{r}^{(\ell)} \in \mathbb{R}^{d_r/c}$ is reshaped into a 2D convolution filter $\mathbf{R}^{(\ell)} \in \mathbb{R}^{h \times w}$. Here, c is the number of filters, h and w the height and width of each filter, and $d_r = chw$. Figure 1(b) gives a simple example of this reshaping process, where a subject vector of length 9 is reshaped into a 3×3 matrix, and a relation vector of length 8 is split and reshaped into

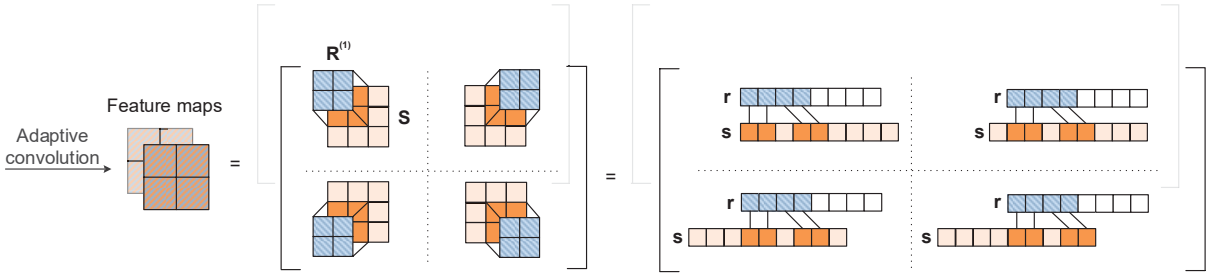


Figure 2: A simple illustration of adaptive convolution, where a 2×2 filter $\mathbf{R}^{(1)}$ (constructed from the first half of the relation vector $\mathbf{r}$) is convolved across a 3×3 input $\mathbf{S}$ (reshaped from the subject vector $\mathbf{s}$), generating a feature map of size 2×2. Each entry of the feature map could be calculated with certain dimensions of $\mathbf{r}$ and $\mathbf{s}$, marked with blue backslash and orange respectively.

two 2×2 filters.[1]

After reshaping, ConvR convolves across the input $\mathbf{S}$ using these adaptively constructed, relation-specific filters. For each filter $\mathbf{R}^{(\ell)}$, a convolutional feature map $\mathbf{C}^{(\ell)} \in \mathbb{R}^{(d_e^h - h + 1) \times (d_e^w - w + 1)}$ will be generated, with the mn-th entry calculated as:

$$c_{m,n}^{(\ell)} = f\left(\sum\nolimits_{i,j} s_{m+i-1,n+j-1} \times r_{i,j}^{(\ell)}\right), \quad (1)$$

where $f(\cdot)$ is a non-linear function, e.g., ReLU (Krizhevsky et al., 2012). Figure 2 visualizes how such a feature map could be generated by convolving across the input with a relation-specific filter (the first equality sign "="), and how each entry of the feature map could be calculated with the original entity and relation vectors (the second equality sign "="). We can see that the adaptive convolution paradigm is quite effective in modeling entity-relation interactions. It enables rich interactions between input entity and relation representations at diverse regions, and all the convolutional features generated will be able to capture such interactions.

Finally, to compute the triple score $\psi(s, r, o)$, we flatten the convolutional feature maps $\mathbf{C}^{(1)}, \cdots, \mathbf{C}^{(c)}$ and stack them into a single vector $\mathbf{c}$. This vector is then projected into $\mathbb{R}^{d_e}$ by a fully-connected layer, and matched with the object vector $\mathbf{o}$ with an inner product, i.e.,

$$\psi(s, r, o) = f(\mathbf{W}\mathbf{c} + \mathbf{b})^\top \mathbf{o}, \quad (2)$$

where $\mathbf{W} \in \mathbb{R}^{d_e \times c(d_e^h - h + 1)(d_e^w - w + 1)}$ and $\mathbf{b} \in \mathbb{R}^{d_e}$ are parameters of the fully-connected layer, and $f(\cdot)$ is again a non-linear function.

[1] During reshaping we consider the most natural ordering of the embedding entries. That means, a length-x vector is reshaped into a $y \times z$ matrix ($x = yz$) such that the first row of the matrix comes from the first z entries of the vector, the second row from the second z entries, and the y-th row from the last z entries.

Parameter learning For learning the model parameters, we follow (Dettmers et al., 2018) and use 1-to-many scoring to speed-up training and evaluation. Unlike traditional 1-to-1 scoring which takes a triple (s, r, o) as input and directly scores it, 1-to-many scoring takes (s, r) as input and scores it against all candidate objects $o \in \mathcal{E}$ simultaneously, generating a score vector $\mathbf{p}^{s,r} \in \mathbb{R}^{|\mathcal{E}|}$. Each dimension of this score vector corresponds to an entity $o \in \mathcal{E}$, calculated as $p_o^{s,r} = \sigma(\psi(s, r, o))$, where $\psi(s, r, o)$ is the triple score defined in Eq. (2) and $\sigma(x) = \frac{1}{1+e^{-x}}$ the sigmoid function. For each input (s, r), we minimize the following cross-entropy loss:

$$\mathcal{L}(s,r) = -\frac{1}{|\mathcal{E}|}\sum_{o \in \mathcal{E}} y_o^{s,r} \log(p_o^{s,r}) + (1 - y_o^{s,r}) \log(1 - p_o^{s,r}), \quad (3)$$

where $y_o^{s,r}$ is a binary label. We have $y_o^{s,r} = 1$ if (s, r, o) is a valid triple and $y_o^{s,r} = 0$ otherwise. During optimization, we use dropout (Srivastava et al., 2014) to prevent overfitting. Specifically, we use dropout on the reshaped subject representations, the convolutional feature maps, and the projected vectors after the fully-connected layer. We also use batch normalization (Ioffe and Szegedy, 2015) on these representations to stabilize and speed up convergence. We use Adam (Kingma and Ba, 2014) optimizer and label smoothing (Szegedy et al., 2016) as suggested by ConvE.

Advantages over ConvE The most prominent advantage of ConvR over ConvE is its high ability to model entity-relation interactions in a convolutional fashion, which is crucial for learning with multi-relational data. ConvE, which convolves across stacked entity-relation representations with global filters, can only model interactions between

the two types of input around the concatenation line, and only a small proportion of the convolutional features would be able to capture such interactions (see Figure 1(a)). ConvR, by contrast, enables entity-relation interactions at diverse regions, and all the convolutional features are able to capture such interactions (see Figure 1(b) and Figure 2).

Besides being more effective, ConvR might potentially be more efficient (in terms of the number of parameters). ConvE has a space complexity of $\mathcal{O}(d|\mathcal{E}| + d|\mathcal{R}| + chw + cd(2d^h - h + 1)(d^w - w + 1))$, where $d|\mathcal{E}|$ is to store entity vectors, $d|\mathcal{R}|$ relation vectors, chw the c global filters with size $h \times w$, $cd(2d^h - h + 1)(d^w - w + 1)$ the projection matrix in the fully-connected layer, and $d = d^h d^w$. As entity and relation representations need to be stacked in ConvE, they are usually of the same size, say d. In ConvR, convolution filters are adaptively constructed from relation vectors, so there is no need for global filters. Also, the input of the convolutional layer will be half-sized, generating smaller feature maps, and hence requires a smaller fully-connected layer. The space complexity of ConvR would be $\mathcal{O}(d_e|\mathcal{E}| + d_r|\mathcal{R}| + cd_e(d_e^h - h + 1)(d_e^w - w + 1))$. Although it could be possible to use different configuration of those common arguments in the two methods (e.g., different number of filters or entity vectors with different size), which may result in different memory cost, we empirically show that ConvR can perform substantially better than ConvE with a good variety of configurations, even those with fewer parameters (see the section "Parameter efficiency of ConvR" for details).

4 Experiments

In this section, we evaluate ConvR against competitive baselines in the link prediction task on multiple benchmark KBs. We also investigate parameter efficiency of ConvR against ConvE to further show its superiority.

4.1 Experimental Setup

Datasets We use four datasets for our experiments. The first two are the popular WN18 and FB15k, both released by (Bordes et al., 2013).[2] WN18 is a subset of WordNet for lexical relationships between words, and FB15k a subgraph of Freebase for generic facts. In most cases WN18 and FB15k encode a relation and its inverse relation at the same time. That means, once a fact is observed, there are usually two distinct triples created for it, e.g., $(s, \text{hyponym}, o)$ and $(o, \text{hypernym}, s)$, or $(s, \text{director-of}, o)$ and $(o, \text{directed-by}, s)$. As pointed out by (Toutanova and Chen, 2015) and (Dettmers et al., 2018), encoding inverse relations might suffer from test leakage, i.e., for each test triple (s, r, o), it is likely to find its inverse (o, r^{-1}, s) in the training set. To avoid this test leakage issue, we further use WN18RR (Dettmers et al., 2018),[3] a subset of WN18 with inverse relations removed, and FB15k-237 (Toutanova and Chen, 2015),[4] a filtered version of FB15k with both inverse and duplicate relations removed. Table 1 summarizes the statistics of the four datasets, where the training sets are used for parameter learning, the validation sets for hyperparameter tuning, and the test sets for evaluation.

Dataset	# Rel	# Ent	# Train	# Valid	# Test
FB15k	1,345	14,951	483,142	50,000	59,071
WN18	18	40,943	141,442	5,000	5,000
FB15k-237	237	14,541	272,115	17,535	20,466
WN18RR	11	40,943	86,835	3,034	3,134

Table 1: Statistics of the four datasets. Columns stand for the number of relations, number of entities, and number of triples in training/validation/test sets.

Evaluation protocol We adopt the ranking process proposed in (Bordes et al., 2013) for evaluation. For each triple (s, r, o) in the test set, we replace the subject s with every entity $e \in \mathcal{E}$, and calculate a score for the corrupted triple (e, r, o). Then we sort these scores in descending order to get the rank of the correct subject s. Since corrupted triples may also be valid, we remove those that already exist in either the training, validation, or test set during ranking, i.e., the *filtered* setting as called by (Bordes et al., 2013). This whole procedure is repeated while replacing the object o. We aggregate over all test triples, and report the mean reciprocal rank (MRR) and the proportion of correct entities ranked in the top n (Hits@n), with $n = 1, 3, 10$.

Implementation details We implement ConvR in PyTorch. In our experiments, we fix mini-batch

[2] https://everest.hds.utc.fr/doku.php?id=en:smemlj12

[3] https://github.com/TimDettmers/ConvE/blob/master/WN18RR.tar.gz

[4] https://www.microsoft.com/en-us/download/details.aspx?id=52312

Dataset	d_e	c	$h \times w$	ρ_1	ρ_2	ρ_3
FB15k	200	100	3×3	0.1	0.4	0.2
WN18	200	100	3×3	0.4	0.3	0.3
FB15k-237	100	100	5×5	0.3	0.2	0.3
WN18RR	200	200	3×3	0.2	0.2	0.5

Table 2: Optimal configurations of ConvR on the four datasets. Columns are the entity embedding size, number and size of filters, and dropout ratios.

size to 128, initial learning rate to 0.001, and label smoothing coefficient to 0.1. Other hyper-parameters are selected with grid search on the validation set. Specifically, we tune the entity embedding size $d_e \in \{100, 200\}$, filter number $c \in \{50, 100, 150, 200\}$, and filter size $h \times w \in \{3 \times 3, 4 \times 4, 5 \times 5\}$. All dropout ratios, i.e., ρ_1 on reshaped subject representations, ρ_2 on convolutional feature maps, and ρ_3 on projected vectors after the fully-connected layer, are tuned in $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. On each dataset, we choose the optimal configuration with the highest MRR on the validation set within 1000 epochs, and report its performance on the test set. Table 2 lists the optimal configurations of ConvR on the four datasets.

Baseline methods We compare ConvR against a variety of competitive baselines, which can be roughly categorized into two groups:

- Methods that use (relatively) simple operations in vector space to model multi-relational data, including TransE (Bordes et al., 2013), DistMult (Yang et al., 2015) and its re-implementation (Kadlec et al., 2017), HolE (Nickel et al., 2016b), ComplEx (Trouillon et al., 2016), ANALOGY (Liu et al., 2017), TorusE (Ebisu and Ichise, 2017), Gaifman (Niepert, 2016), KBGAN (Cai and Wang, 2017), KBLRN (Garcia-Duran and Niepert, 2018), and Node+LinkFeat (Toutanova and Chen, 2015).

- Methods that further introduce multi-layer structures and non-linearity, in particular those based on neural networks, including R-GCN (Schlichtkrull et al., 2017a), Neural LP (Yang et al., 2017), ConvE (Dettmers et al., 2018), and ConvKB (Nguyen et al., 2018).

4.2 Link Prediction Results

Table 3 reports the results on WN18 and FB15k, and Table 4 the results on WN18RR and FB15k-237. On all the four datasets, the results for the baselines are taken directly from previous literature to avoid re-implementation bias. Since not all baselines have their results reported on all the four datasets, we cannot make the two sets of baselines compared in Table 3 and Table 4 exactly the same. From the results, we can see that: (1) On WN18 and FB15k, ConvR performs better than or at least as well as the baselines in almost all the metrics. (2) Compared to ConvE, it offers a 5% increase in MRR, a 7% increase in Hits@1, and a 2% increase in Hits@10 on FB15k. (3) On the more difficult WN18RR and FB15k-237, ConvR consistently outperforms most of the baselines, except for MRR score of ConvKB on FB15k-237. However, on WN18RR ConvR outperforms ConvKB on all known metrices, especially MRR. This discrepancy may be attributed to ConvKB's initialization with TransE on FB15k-237. (4) Compared to ConvE, it offers a 3% increase in MRR, a 14% increase in Hits@1, a 12% increase in Hits@10 on WN18RR, and an 11% increase in MRR, a 9% increase in Hits@1, an 8% increase in Hits@10 on FB15k-237.

4.3 Parameter Efficiency of ConvR

We further investigate parameter efficiency of ConvR against ConvE on FB15k-237. Specifically, we tune the number of filters $c \in \{20, 40, 60, 80, 100\}$ and the filter size $h \times w \in \{2 \times 2, 3 \times 3, 4 \times 4, 5 \times 5\}$, fix the other hyper-parameters to their optimal configurations (see Table 2 for details), and show how the performance of ConvR (on the test set) will change as the number of parameters varies. For comparison, we directly show the performance and parameter efficiency of the optimal ConvE model, as reported in (Dettmers et al., 2018). The results are given in Table 5.[5]

From the results, we can see that: (1) The parameter number of ConvR steadily grows as the filter number c and filter size $h \times w$ increase, but the performance does not change much. That means, ConvR might achieve relatively good (though not best) performance with a potentially small number

[5]Note that some results reported here are even better than those reported in Table 4. This is because in Table 4 we determine optimal configurations according to MRR on the validation set, which may not necessarily lead to best performance on the test set.

	WN18				FB15k			
	MRR	Hits @1	Hits @3	Hits @10	MRR	Hits @1	Hits @3	Hits @10
TransE (Bordes et al., 2013) †	0.454	0.089	0.823	0.934	0.380	0.231	0.472	0.641
DistMult (Yang et al., 2015) †	0.822	0.728	0.914	0.936	0.654	0.546	0.733	0.824
DistMult(Kadlec et al., 2017)	0.797	–	–	0.946	0.798	–	–	**0.893**
HolE (Nickel et al., 2016b)	0.938	0.930	0.945	0.949	0.524	0.402	0.613	0.739
ComplEx (Trouillon et al., 2016)	0.941	0.936	0.945	0.947	0.692	0.599	0.759	0.840
ANALOGY (Liu et al., 2017)	0.942	0.939	0.944	0.947	0.725	0.646	0.785	0.854
TorusE (Ebisu and Ichise, 2017)	0.947	0.943	0.950	0.954	0.733	0.674	0.771	0.832
Gaifman (Niepert, 2016)	–	0.761	–	0.939	–	0.692	–	0.842
KBLRN (Garcia-Duran and Niepert, 2018)	–	–	–	–	0.794	**0.748**	–	0.875
Node+LinkFeat (Toutanova and Chen, 2015)	0.940	–	–	0.943	**0.822**	–	–	0.870
R-GCN (Schlichtkrull et al., 2017b)	0.814	0.686	0.928	0.955	0.651	0.541	0.736	0.825
Neural LP (Yang et al., 2017)	0.94	–	–	0.945	0.76	–	–	0.837
ConvE (Dettmers et al., 2018)	0.942	0.935	0.947	0.955	0.745	0.670	0.801	0.873
ConvR (this work)	**0.951**	**0.947**	**0.955**	**0.958**	0.782	0.720	**0.826**	0.887

Table 3: Link prediction results on the test sets of WN18 and FB15k. Results marked by † are taken from (Trouillon et al., 2016). Other results are taken from the original papers. Missing scores not reported are denoted by "–". Best scores highlighted in bold.

	WN18RR				FB15k-237			
	MRR	Hits @1	Hits @3	Hits @10	MRR	Hits @1	Hits @3	Hits @10
DistMult (Yang et al., 2015) ‡	0.43	0.39	0.44	0.49	0.241	0.155	0.263	0.419
ComplEx (Trouillon et al., 2016) ‡	0.44	0.41	0.46	0.51	0.247	0.158	0.275	0.428
KBGAN (Cai and Wang, 2017)	0.214	–	–	0.472	0.278	–	–	0.458
KBLRN (Garcia-Duran and Niepert, 2018)	–	–	–	–	0.309	0.219	–	0.493
Node+LinkFeat (Toutanova and Chen, 2015)	–	–	–	–	0.226	–	–	0.347
R-GCN (Schlichtkrull et al., 2017b)	–	–	–	–	0.248	0.153	0.258	0.417
Neural LP (Yang et al., 2017)	–	–	–	–	0.24	–	–	0.362
ConvE (Dettmers et al., 2018)	0.46	0.39	0.43	0.48	0.316	0.239	0.350	0.491
ConvKB (Nguyen et al., 2018)	0.248	–	–	0.525	**0.396**	–	–	0.517
ConvR (this work)	**0.475**	**0.443**	**0.489**	**0.537**	0.350	**0.261**	**0.385**	**0.528**

Table 4: Link prediction results on the test sets of WN18RR and FB15k-237. Results marked by ‡ are taken from (Dettmers et al., 2018). Other results are taken from the original papers. KBGAN refers to the "TransD + DistMult" setting which shows best performance. ConvKB is initialized with TransE on FB15k-237, and randomly on WN18RR. Missing scores not reported are denoted by "–". Best scores highlighted in bold.

of parameters. (2) ConvR consistently and substantially outperforms the best performing ConvE with all the configurations listed in Table 5. (3) In particular, even the most efficient configuration (i.e., $c = 20$ and $h \times w = 2 \times 2$) offers a 7% increase in MRR and a 6% increase in Hits@10, with its parameter number only 88% as large as that of ConvE.

5 Related Work

Link prediction is a crucial task for knowledge bases (KBs). A good variety of statistical relational learning techniques have been proposed for this task (Nickel et al., 2016a), among which vector space embedding models are most particular due to their superior performance and potential scalability. Early works of this kind tend to employ simple vector space operations for link prediction. For example, TransE (Bordes et al., 2013) takes relations as translations between subject and object entities. DistMult (Yang et al., 2015) uses multilinear dot product to characterize three-way interactions among subjects, relations, and objects. ComplEx (Trouillon et al., 2016) further generalizes DistMult to complex vector space. Using simple models allows one to easily handle large-scale KBs, but usually at the cost of less model expressiveness (Dettmers et al., 2018). HolE (Nickel et al., 2016b) tries to increase model expressiveness while keeping simplicity. It uses cross-correlation, i.e., the

	$h \times w = 2 \times 2$	$h \times w = 3 \times 3$	$h \times w = 4 \times 4$	$h \times w = 5 \times 5$
ConvE	–	1.89M | 0.32 | 0.49	–	–
ConvR, $c = 20$	1.67M | 0.342 | 0.520	1.68M | 0.342 | 0.522	1.72M | 0.342 | 0.522	1.78M | 0.342 | 0.522
ConvR, $c = 40$	1.87M | 0.345 | 0.526	1.90M | 0.344 | 0.524	1.97M | 0.348 | 0.529	2.09M | 0.347 | 0.526
ConvR, $c = 60$	2.07M | 0.347 | 0.525	2.11M | 0.348 | 0.530	2.22M | 0.350 | 0.529	2.40M | 0.350 | 0.527
ConvR, $c = 80$	2.27M | 0.347 | 0.528	2.32M | 0.348 | 0.532	2.47M | 0.350 | 0.532	2.71M | 0.348 | 0.528
ConvR, $c = 100$	2.47M | 0.348 | 0.531	2.54M | 0.348 | 0.527	2.72M | 0.349 | 0.527	3.02M | 0.350 | 0.528

Table 5: Parameter efficiency on FB15k-237. Each cell reports number of parameters, MRR, and Hits@10 in turn. Results of ConvE are taken from (Dettmers et al., 2018).

inverse of circular convolution, to match subject and object entities, which has some similarity to our work. But HolE is not a typical neural network architecture. It does not learn multiple layers of non-linear features, and hence is less expressive than our approach.

A more direct way of increasing model expressiveness is to employ deeper, more complicated neural network architectures, e.g., multi-layer perceptron (Dong et al., 2014), semantic matching energy networks (Bordes et al., 2014), and neural tensor networks (Socher et al., 2013). This kind of approaches, however, often have more parameters and are prone to overfit (Nickel et al., 2016a). (Dettmers et al., 2018) recently devised ConvE, a multi-layer convolutional network which offers increased model expressiveness while remaining highly parameter efficient. After that, (Nguyen et al., 2018) propose ConvKB that explores the global relationships among same dimensional entries of the entity and relation embeddings. However, neither of them models the interactions between various positions of entities and relations. R-GCN (Schlichtkrull et al., 2017a) is another convolutional network designed for KBs, generalized from GCN (Kipf and Welling, 2016) for uni-relational data. But the convolution of R-GCN is conducted in a message passing manner, quite different from our work.

Convolutional neural networks have been successfully applied to a wide variety of domains, ranging from speech or visual recognition (Abdel-Hamid et al., 2014; Krizhevsky et al., 2012) to natural language processing (Collobert et al., 2011). Similar ideas of using adaptive or dynamic convolutional filters have been studied before (Lee et al., 2010; Jia et al., 2016; Kang et al., 2017). But most of such works focus on image or video processing. This work focuses on multi-relational data and devises an adaptive convolution paradigm particularly suitable for this scenario.

6 Conclusion

In this paper, we propose ConvR, an adaptive convolutional network specially designed for learning with multi-relational data. In contrast to previous work which convolves across stacked representations with global filters, ConvR adaptively constructs convolution filters from relation representations, and applies these filters across entity representations to generate convolutional features. This adaptive convolution paradigm enables rich interactions between entity and relation representations at diverse regions, and all convolutional features generated in this way will be able to capture such interactions. Experimental results on multiple benchmark knowledge bases show that ConvR achieves significant and consistent improvements against a variety of baselines. In particular, it is not only more effective but also more efficient than state-of-the-art convolutional models, offering a 7% increase in MRR and a 6% increase in Hits@10, while saving 12% in parameter storage.

As future work, we plan to devise convolutional paradigms that can maximize interactions not only between subject entities and relations, but also between object entities and relations. In ConvR, we use 1-to-many scoring to speed up training and evaluation. As a side effect, object representations can only interact with a hidden vector (output of the fully-connected layer) via an inner product, which potentially limits the performance of ConvR. It is worth investigating modeling these interactions while keeping the merit of fast training and evaluation.

Acknowledgments

We would like to thank all the anonymous reviewers for their insightful and valuable suggestions, which help to improve the quality of this paper. This work is supported by the National Natural Science Foundation of China (grant No. 61876223).

References

Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. 2014. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(10):1533–1545.

Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259.

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*, pages 2787–2795.

Liwei Cai and William Yang Wang. 2017. KBGAN: adversarial learning for knowledge graph embeddings. *arXiv preprint arXiv:1711.04071*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. 2018. Convolutional 2D knowledge graph embeddings. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 1811–1818.

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 601–610.

Takuma Ebisu and Ryutaro Ichise. 2017. TorusE: Knowledge graph embedding on a lie group. *arXiv preprint arXiv:1711.05435*.

Alberto García-Durán, Antoine Bordes, and Nicolas Usunier. 2014. Effective blending of two and three-way interactions for modeling multi-relational data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 434–449.

Alberto Garcia-Duran and Mathias Niepert. 2018. Kblrn: End-to-end learning of knowledge base representations with latent, relational, and numerical features. In *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence (UAI)*.

Lise Getoor and Ben Taskar. 2007. *Introduction to statistical relational learning*. MIT press.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Rodolphe Jenatton, Nicolas L. Roux, Antoine Bordes, and Guillaume R. Obozinski. 2012. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems*, pages 3167–3175.

Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. 2016. Dynamic filter networks. In *Advances in Neural Information Processing Systems*, pages 667–675.

Rudolf Kadlec, Ondrej Bajgar, and Jan Kleindienst. 2017. Knowledge base completion: Baselines strike back. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 69–74.

Di Kang, Debarun Dhar, and Antoni Chan. 2017. Incorporating side information by adaptive convolution. In *Advances in Neural Information Processing Systems*, pages 3870–3880.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105.

Kyong Joon Lee, Dongjin Kwon, Il Dong Yun, and Sang Uk Lee. 2010. Optical flow estimation with adaptive convolution kernel prior on discrete framework. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2504–2511.

Hanxiao Liu, Yuexin Wu, and Yiming Yang. 2017. Analogical inference for multi-relational embeddings. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2168–2178.

Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2018. A novel embedding model for knowledge base completion based on convolutional neural network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 327–333.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2016a. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33.

Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, et al. 2016b. Holographic embeddings of knowledge graphs. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 1955–1961.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on Machine Learning*, pages 809–816.

Mathias Niepert. 2016. Discriminative gaifman models. In *Advances in Neural Information Processing Systems*, pages 3405–3413.

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2017a. Modeling relational data with graph convolutional networks. *arXiv:1703.06103*.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2017b. Modeling relational data with graph convolutional networks. *arXiv preprint arXiv:1703.06103*.

Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pages 926–934.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and Their Compositionality*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 2071–2080.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 1112–1119.

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

Fan Yang, Zhilin Yang, and William W Cohen. 2017. Differentiable learning of logical rules for knowledge base reasoning. In *Advances in Neural Information Processing Systems*, pages 2316–2325.