The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Incorporating GAN for Negative Sampling in Knowledge Representation Learning

Peifeng Wang
School of Data and Computer Science,
Sun Yat-sen University, China.
wangpf3@mail2.sysu.edu.cn

Shuangyin Li
iPIN Inc., Shenzhen, China.
shuangyinli@ipin.com

Rong Pan*
School of Data and Computer Science,
Sun Yat-sen University, China.
panr@sysu.edu.cn

Abstract

Knowledge representation learning aims at modeling knowledge graph by encoding entities and relations into a low dimensional space. Most of the traditional works for knowledge embedding need negative sampling to minimize a margin-based ranking loss. However, those works construct negative samples through a random mode, by which the samples are often too trivial to fit the model efficiently. In this paper, we propose a novel knowledge representation learning framework based on Generative Adversarial Networks (GAN). In this GAN-based framework, we take advantage of a generator to obtain high-quality negative samples. Meanwhile, the discriminator in GAN learns the embeddings of the entities and relations in knowledge graph. Thus, we can incorporate the proposed GAN-based framework into various traditional models to improve the ability of knowledge representation learning. Experimental results show that our proposed GAN-based framework outperforms baselines on triplets classification and link prediction tasks.

1 Introduction

Knowledge graph refers to a network whose nodes are entities in the real world and edges are relations between entities. Such a network builds up a structural system for human knowledge and is composed of a large number of facts in the form of triplet (head entity, relation, tail entity). Knowledge graph plays an important role in supporting applications such as web search, question answering, and personalized recommendation. Many knowledge graphs, such as Freebase (Bollacker et al. 2008), DBpedia (Auer et al. 2007), and YAGO (Suchanek, Kasneci, and Weikum 2007), have already been well-developed. Lots of previous works (Bordes et al. 2013; Wang et al. 2014; Lin et al. 2015; He et al. 2015) on knowledge graphs make great efforts in knowledge representation learning to handle data sparsity and incompleteness. Knowledge representation learning attempts to embed a knowledge graph into a continuous vector space. It provides a numerical computation framework for knowledge graph completion and captures semantic similarity between entities, which is useful for solving data sparsity.

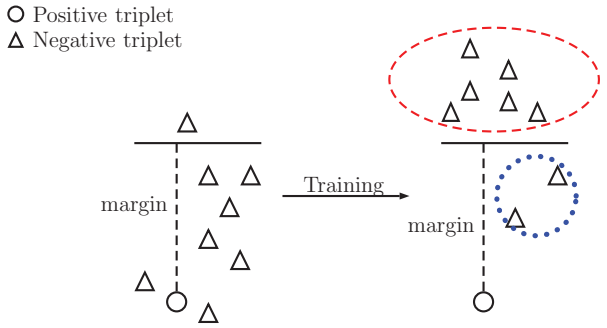

Figure 1: Illustration of zero loss problem in the random mode. After training for a while, the random mode often samples triplets in the red segmented circle, which are out of the margin and bring no loss. A certain number of negative triplets are thus ignored within the margin as shown in the blue dotted circle.

To date, many knowledge representation learning models have been proposed and have shown to achieve great successes for their efficiency. These models try different approaches to score a triplet (h, r, t), which assume that there are positive triplets from the training set and man-made negative triplets. The target of these models is to minimize a margin-based ranking loss which aims to separate the scores of the positive triplets from those of the negative triplets by a fixed margin.

Nevertheless, the negative triplets are constructed by replacing either head or tail entities of positive triplets with entities randomly sampled from the whole entities set. Such a random sampling method is effective at the beginning of the training since most of the negative triplets are still within the margin to the positive triplets. However, when the training process goes on, if we still construct negative triplets by random sampling, they are very likely to be out of the margin, which gives rise to zero loss problem as illustrated in Figure 1. Thus, these trivial negative triplets fail to provide guidances on model training.

Moreover, a high-quality negative triplet can help push the model to its limit. Take the triplet (*Steve Jobs, FounderOf, Apple Inc.*) as an example. We would like to replace its head

*Corresponding author.

entity, *Steve Jobs*, to generate a negative triplet. If we just sample randomly, there is a large probability that we might sample some less informative negative entities such as *London* or *baseball* which are not person entity at all, resulting zero loss easily. Being told over and over again that *Steve Jobs* is different from the non-person entities, the model can not learn the real concept of this person entity. Thus randomly generated negative triplets usually do little help in training the true triplets and even slows down the convergence as pointed out by (Schroff, Kalenichenko, and Philbin 2015; Hermans, Beyer, and Leibe 2017). Therefore, it is necessary to mine quality negative triplets while training.

Thus, in this paper, we propose a knowledge embedding framework based on GAN (Goodfellow et al. 2014) to improve the negative sampling in knowledge representation learning. The discriminator in it is trained to minimize the margin-based ranking loss as in the previous models. The generator learns mining quality negative samples which can bring non-zero loss to the discriminator. The embeddings learned by the discriminator are used as the final representation of the knowledge graph. The generator could be considered as an auxiliary which pushes the discriminator to its limit in modeling the knowledge graph. Our contributions in this paper are as follows.

1. We incorporate the GAN framework in knowledge representation learning, and prove that the generator can consistently provide quality negative triplets which is crucial for the discriminator to minimize the margin-based ranking loss.
2. The proposed GAN-based training framework can be easily extended to various knowledge representation learning models and enhance their ability in knowledge embedding tasks.

We evaluate the proposed GAN-based training framework on two knowledge graph related tasks, triplets classification and link prediction. The experimental results show that the performances of our GAN-based models can achieve remarkable improvements over the baselines, which justifies the effectiveness of our model in mining quality negative samples and embedding knowledge graph.

2 Related Work

For symbol clarification, we define an entity set $\mathbf{E}$ and a relation set $\mathbf{R}$. We denote a triplet as (h, r, t) where h, t represent head and tail entity respectively, and r indicates the relation between h and t. The corresponding embeddings we would like to learn are $\mathbf{h}$, $\mathbf{t}$ and $\mathbf{r}$ for entities h, t and relation r.

TransE (Bordes et al. 2013) starts the line of translation based knowledge embedding models and the basic idea of these models is that if a triplet (h, r, t) holds, then it represents a translation from a head entity to a tail entity via a relation vector, namely $\mathbf{h} + \mathbf{r} = \mathbf{t}$. The score function used by TransE for a triplet (h, r, t) is defined as

$$f_r(h,t) = |\mathbf{h} + \mathbf{r} - \mathbf{t}|_{L1/L2}, \tag{1}$$

where L1 and L2 represent L1 and L2 norm. The model should achieve lower score if a triplet holds.

Since TransE is not suitable for 1-to-N, N-to-1 or N-to-N relations, **TransH** (Wang et al. 2014) proposes to project the entity to the relation specific hyperplane $\mathbf{w_r}$ and thus an entity could have different representations depending on what relation evolved. In detail, TransH transforms the entity embeddings as

$$\mathbf{h}_\perp = \mathbf{h} - \mathbf{w_r}^\top \mathbf{h} \mathbf{w_r}, \quad \mathbf{t}_\perp = \mathbf{t} - \mathbf{w_r}^\top \mathbf{t} \mathbf{w_r}, \tag{2}$$

and the score function is defined as

$$f_r(h,t) = |\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp|_{L1/L2}. \tag{3}$$

Both TransE and TransH assume that entities and relations are in the same vector space, thus **TransR** (Lin et al. 2015) proposes to use a relation specific matrix $\mathbf{M_r}$ to project the entity embedding into the relation vector space and the score function becomes

$$f_r(h,t) = |\mathbf{M_r h} + \mathbf{r} - \mathbf{M_r t}|_{L1/L2}. \tag{4}$$

TransD (Ji et al. 2015) argues that the project matrix should not only be relation specific, but also be determined by entities themselves. Therefore the project matrices are defined as

$$\mathbf{M_{rh}} = \mathbf{r_p h_p}^\top + \mathbf{I}, \quad \mathbf{M_{rt}} = \mathbf{r_p t_p}^\top + \mathbf{I}, \tag{5}$$

where subscript p marks the projection vectors.

Other translation embedding models project entities in different ways like (Xiao, Huang, and Zhu 2015; Ji et al. 2016) or make use of supplement information including text descriptions (Zhong et al. 2015; Xie et al. 2016b; Xu et al. 2017) and images (Xie et al. 2016a) of entities.

Unstructured Model (Bordes et al. 2012; 2014) ignores the relation between entities and the score function is defined as

$$h_r(h,t) = |\mathbf{h} - \mathbf{t}|. \tag{6}$$

Structured Embedding (Bordes et al. 2011) uses two projection matrix $\mathbf{M_{r,h}}$ and $\mathbf{M_{r,t}}$ to represent a relation r. The score function is defined as

$$f_r(h,t) = |\mathbf{M_{r,h}h} - \mathbf{M_{r,t}t}|. \tag{7}$$

Single Layer Model (Socher et al. 2013) improves Structured Embedding Model by defining additional relation vector $\mathbf{u_r}$ for each relation and nonlinear transformation. Its score function for a triplet is

$$f_r(h,t) = -\mathbf{u_r}^\top g(\mathbf{M_{r,h}h} + \mathbf{M_{r,t}t} + \mathbf{b_r}), \tag{8}$$

where g() is tanh function.

Neural Tensor Network (Socher et al. 2013) extends Single Layer Model by considering the second-order correlations into non-linear transformation. Its score function is defined as

$$f_r(h,t) = -\mathbf{u_r}^\top g(\mathbf{h}^\top \mathbf{M_r t} + \mathbf{M_{r,h}h} + \mathbf{M_{r,t}t} + \mathbf{b_r}), \tag{9}$$

where $\mathbf{M_r}$ is a three-way tensor.

All the above knowledge embedding models construct negative triplets by random sampling and are trained to separate the scores between positive and negative triplets. In this paper, we propose to incorporate GAN training framework for better negative sampling. GAN has been applied

to many others scenarios, such as selecting difficult documents in information retrieval (Wang et al. 2017) and generating fake sentences for commonsense machine comprehension (Wang, Liu, and Zhao 2017). We are the first to employ GAN on knowledge representation learning for high-quality negative triplets sampling.

3 GAN on Knowledge Embedding

Given a training set S of triplets (h, r, t), we try to learn the embeddings of entities and relations. Consider there's a positive triplet (h, r, t) that holds in knowledge graph and a negative triplet $(h^{'}, r, t^{'})$ that we need to construct. To ensure that the model gives a low score if a triplet (h, r, t) holds and a high score otherwise, the margin-based ranking loss is considered as

$$l(h, r, t) = [f_r(h, t) - f_r(h^{'}, t^{'}) + \gamma]_+, \tag{10}$$

where $[x]_+ = max\{0, x\}$ denotes the positive part of x, and γ is a fixed margin set as hyper parameter. The score function $f_r(h, t)$ has been described in Section 2. The objective function of the model is defined as

$$\min \sum_{(h,r,t)\in\Delta} \sum_{(h^{'},r,t^{'})\in\Delta^{'}} l(h, r, t), \tag{11}$$

where the Δ denotes the positive triplets set from the knowledge graph and $\Delta^{'}$ indicates the negative triplets set constructed by replacing the head or tail entity of the triplets in Δ by other entity in $\mathbf{E}$, namely

$$\Delta^{'} = \{(h^{'}, r, t)|h^{'} \in \mathbf{E}\} \cup \{(h, r, t^{'})|t^{'} \in \mathbf{E}\}. \tag{12}$$

In practice, the model enforces constrains as $\|\mathbf{h}\|_2 \leq 1$, $\|\mathbf{r}\|_2 \leq 1$ and $\|\mathbf{t}\|_2 \leq 1$, where $\|\cdot\|_2$ refers to L2 norm.

During training, positive triplets are traversed randomly. When a positive triplet is visited, previous methods generate a negative triplet by replacing the head entity or tail entity (not both at the same time) with an entity sampled randomly from the whole entity set $\mathbf{E}$. There are two strategies to decide whether to replace head or tail entity, (1) "**unif**" method which replaces head or tail with equal probability; (2) "**bern**" method from (Wang et al. 2014) which replaces head or tail according to Bernoulli distribution.

Zero loss problem. The negative triplets generated by random sampling are effective at the beginning of training. However, after training for a while, most of these randomly generated negative triplets achieve scores that are out of the margin to the positive triplets and would lead to zero loss in Eq. (10) trivially. These negative triplets make no contribution to improving the embeddings. Thus such a sampling method would cause very slow convergence and even fail to get the best result.

To generate quality negative triplets and accelerate the training process, we propose to incorporate the GAN framework in our model, under which the discriminator is trained to minimize the margin-based ranking loss in Eq. (10) as the previous models, while the generator learns to continually generate quality negative triplets that would not lead to zero loss. We describe the discriminator and generator respectively in Section 3.1 and Section 3.2.

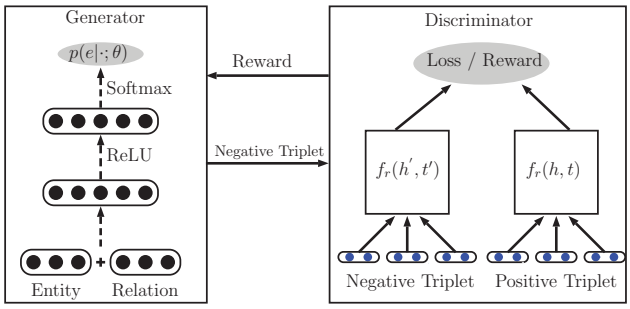

Figure 2: GAN-based training framework. (a) Generator learns to provide quality negative triplets according to the reward from discriminator. (b) Discriminator learns to embed the knowledge graph using the negative triplets from generator.

3.1 Discriminator for Knowledge Embedding

The discriminator in our model is designed as the previous models. For different models, the score function is computed in various ways as introduced in Section 2. Then the embeddings of entities and relations are learned by training the discriminator to minimize the margin-based ranking loss between positive triplets and negative triplets as defined in Eq. (10). Different to the previous models whose negative triplets are constructed by randomly sampling from the whole entity set $\mathbf{E}$, the discriminator uses the negative triplets generated by the generator which will be described in detail as in Section 3.2.

3.2 Generator for Negative Sampling

The goal of the generator is to provide the discriminator with quality negative triplets which could bring non-zero loss. Note that, the generator in our model has its own embeddings (different from discriminator's) for entities and relations. Besides, there are two separate embeddings for each relation r, one for normal r as usual and another for the reverse of the relation, namely r^{-1}. For each positive triplet (h, r, t), the input of the generator is an entity-relation pair defined as

$$\begin{cases} (t, r), & z = 1 \\ (h, r^{-1}), & z = 0, \end{cases} \tag{13}$$

where $z \in \{1, 0\}$ is a binary flag indicating whether to replace head entity or tail entity, which is decided by either "**unif**" or "**bern**" strategy.

The embeddings of the entity-relation pair are concatenated and fed to a two layers fully-connected neural network as shown in the "Generator" part of Figure 2. Nonlinear function ReLU is added after the first layer and Softmax function is added to the second layer. The network and the generator's embedding matrices for entities and relations are used to parametrize the probability distribution over the whole entity set $\mathbf{E}$. The probability distribution of the entity set $\mathbf{E}$ is defined as

$$\begin{aligned} p(e|(h, r, t), z; \theta) = z \cdot p(e|t, r; \theta) \\ + (1 - z) \cdot p(e|h, r^{-1}; \theta), \end{aligned} \tag{14}$$

from which we sample an entity to be the replacement. Since the output of the generator is a discrete index of the entity, we use policy gradient based reinforcement learning (Williams 1992; Sutton et al. 2000) and thus the whole network could be viewed as the policy network in the field of reinforcement learning. For a newly generated negative triplet $(h^{'}, r, t)$ or $(h, r, t^{'})$ together with the positive triplet (h, r, t), the reward function calculating by the discriminator is defined as

$$R = tanh(f_r(h, t) - f_r(h^{'}, t^{'}) + \gamma), \tag{15}$$

where the inner part of the activation function $tanh(\cdot)$ is similar to the ranking loss used in Eq. (10) except that the negative part is kept. Such a reward would be positive when the generated negative triplet together with the corresponding positive triplet achieve a non-zero loss in Eq. (10) and negative otherwise.

The generator is trained to maximize the expected reward

$$J(\theta) = E_{e\sim p(e|\cdot;\theta)}[R], \tag{16}$$

from which we could see that in order to achieve higher reward, the policy used by the generator would punish the trivial negative entities by lowering down their corresponding probability and encourage the network to distribute more probability to the entities that can bring non-zero loss.

The generator is updated using the policy gradient as

$$\nabla_\theta J(\theta) = \nabla_\theta \log p(e|(h, r, t), z; \theta) \cdot R. \tag{17}$$

3.3 Model Training

The training process of our proposed model is summarized in Algorithm 1. To be specific, at each training epoch we iterate over the training set in mini-batch to train the generator while the parameters of the discriminator are fixed. Then we iterate over the training set again to train the discriminator while the parameters of the generator are fixed. This training process could also be regarded as a way to make the generator search for the quality negative triplets and filter out the less informative ones before we use it to challenge the discriminator. Both the discriminator and generator are trained by Adam stochastic optimization (Kingma and Ba 2014). In addition, we apply L_2 regularization on generator's parameters, namely θ. When the model converges, we take the embeddings learned by the discriminator as our final representation for entities and relations.

4 Experiments and Analysis

Experiments are conducted on two knowledge graphs, Freebase (Bollacker et al. 2008) and Wordnet (Miller 1995) for two reasoning tasks, link prediction and triplets classification. Freebase contains common facts of the world like (*louis_duke_of_brittany, place_of_birth, france*), and we use two subsets of Freebase. They are FB15K used in (Bordes et al. 2013) and FB13 used in (Socher et al. 2013). Wordnet provides lexical relations between words like (*_discuss_2, _type_of, _talk_about_2*) and we also use two subsets of Wordnet. They are WN11 used in (Socher et al. 2013) and WN18 used in (Bordes et al. 2013). The statistics are listed in Table 1.

Algorithm 1 GAN for knowledge representation learning

Input: Generator G, Discriminator D;
Train set $\mathcal{S} = \{(h, r, t)\}$
Output: Knowledge representations learned by D

```
loop
  for g steps do
    Sample a batch of positive triplets from S
    Use G to generate negative triplets
    Use D to calculate REWARD in Eq. (15)
    Update G's parameters via policy gradient in Eq. (17)
  end for
  for d steps do
    Sample a batch of positive triplets from S
    Use G to generate negative triplets
    Use D to calculate LOSS in Eq. (10)
    Update D's parameters w.r.t Eq. (11)
  end for
end loop
```

Table 1: Data sets used in the experiments.

Dataset	#Rel	#Ent	#Train	#Valid	#Test
FB15K	1,345	14,951	483,142	50,000	59,071
FB13	13	75,043	316,232	5,908	23,733
WN11	11	38,696	112,581	2,609	10,544
WN18	18	40,943	141,442	5,000	5,000

We compare the proposed GAN-based knowledge embedding framework with the models introduced in Section 2. Two training settings are conducted as follows.

- **GAN-scratch** The parameters of the discriminator and generator are initialized randomly. Thus the whole model is trained from scratch.
- **GAN-pretrain** We firstly train the original models with random negative sampling. Then the embeddings learned by the models are used to initialize the parameters of the discriminator. The parameters of the generator are still initialized randomly. Thus the whole model is trained to fine-tune the original models.

4.1 Link Prediction

Link prediction aims to predict the missing entity in a triplet, namely to predict h given (r, t) or t given (h, r). It reflects the model's ability to do knowledge reasoning based on the embeddings it learns. Instead of predicting the golden entity, we list the ranked candidate entities for each incomplete triplet. The experiments are conducted on the data sets WN18 and FB15K. The models we would like to enhance with GAN are listed in Table 2.

For each test triplet (h, r, t), we replace the head (or tail) entity by all the entities in the entities set E. Then we compute the score function $f_r(h, t)$ for the test triplet and its corresponding corrupted triplets. Next we rank the scores in a descending order. We report two measures as the previous studies on knowledge representation learning, the aver-

Table 2: Evaluation results on link predictions.

FB15K	Original		GAN-scratch		GAN-pretrain	
	Mean Rank	Hits@10 (%)	Mean Rank	Hits@10 (%)	Mean Rank	Hits@10 (%)
Unstructured (Bordes et al. 2012)	979	6.3	332	24.5	**305**	**28.7**
SE (Bordes et al. 2011)	**162**	39.8	207	53.3	220	**53.8**
SME(Bilinear) (Bordes et al. 2012)	158	41.3	**139**	**45.8**	170	45.7
TransE (Bordes et al. 2013)	**61**	69.6	90	73.1	81	**74.0**
TransH (Wang et al. 2014)	87	64.4	90	73.3	**81**	**77.0**
TransR (Lin et al. 2015)	**77**	68.7	138	58.3	86	**76.0**
TransD (Ji et al. 2015)	91	77.3	89	74.0	**79**	**77.6**
TransE + A-LSTM (Xu et al. 2017)	77	75.5	79	75.8	**74**	**76.3**
WN18	Original		GAN-scratch		GAN-pretrain	
	Mean Rank	Hits@10 (%)	Mean Rank	Hits@10 (%)	Mean Rank	Hits@10 (%)
Unstructured (Bordes et al. 2012)	**304**	38.2	477	80.5	431	**82.9**
SE (Bordes et al. 2011)	985	80.5	773	86.1	**762**	**87.4**
SME(Bilinear)	509	61.3	**358**	74.6	417	**78.9**
TransE (Bordes et al. 2013)	251	89.2	244	**92.7**	**240**	91.3
TransH (Wang et al. 2014)	303	86.7	276	86.9	**258**	**94.0**
TransR (Lin et al. 2015)	225	92.0	**213**	87.3	291	**93.7**
TransD (Ji et al. 2015)	229	92.5	**221**	93.0	248	**93.3**
TransE + A-LSTM (Xu et al. 2017)	123	90.9	**114**	92.3	120	**92.7**

age rank of the correct entities (Mean Rank) and the proportion of the correct entities ranked in top 10 (Hits@10). Lower Mean Rank or higher Hits@10 reflects better results. Since a corrupted triplet might also exist in the knowledge graph, ranking it ahead of the original triplet is also acceptable. Thus we follow previous studies and filter out all the corrupted triplets that exist in train, validation or test set in order to avoid underestimating the performance of our proposed model, which is called as "Filter" setting.

Implementation. We use Adam SGD for optimization. We select the margin γ among $\{0.5, 1.0, 2.0, 4.0\}$, the dimension d of entities and relations among $\{20, 50, 100, 200\}$, learning rate λ for SGD among $\{0.01, 0.001, 0.0001, 0.00005\}$ and the batch size B among $\{512, 1024, 2048, 4096\}$. We search the best configuration according to the performance of the model in validation set. For "GAN-scratch", the optimal configurations are $\gamma = 2.0$, $d = 100$, $\lambda = 0.001$, and $B = 1024$ on WN18; $\gamma = 1.0$, d = 100, $\lambda = 0.0001$, and $B = 4096$ on FB15K. For "GAN-pretrain", the optimal configurations are $\gamma = 2.0$, d = 100, $\lambda = 0.00005$, and $B = 1024$ on WN18; $\gamma = 1.0$, $d = 100$, $\lambda = 0.0001$, and $B = 2046$ on FB15K. For both training settings, wo adopt the L1 as dissimilarity and use the "unif" strategy to decide whether to replace head or tail during training.

Result. Evaluation results are reported in Table 2. Since the datasets and the evaluation protocol are the same, we choose to reprint the best "Filter" experimental results of the original models directly from the literature, which are referred as "Original" in Table 2, if our own implementations do not yield better results than the ones reported in the literature. Two training settings of our proposed model are referred as "GAN-scratch" and "GAN-pretrain" respectively. We can conclude from Table 2 as follows.

1. For most of the knowledge embedding models, our proposed GAN training framework yields a better performance on FB15K and WN11 respectively. This illustrates the effectiveness of our GAN framework for knowledge representation learning, due to the fact that the generator could generate more quality negative triplets than random sampling does. These non-trivial negative triplets push the discriminator to better rank the correct positive triplets. We also justify this point by visualizing the negative triplets generated by our generator in Section 4.4.
2. Most of the models trained under the "GAN-pretrain" setting perform better than their counterparts trained under the "GAN-scratch" setting. This phenomenon is also observed in triplets classification task, which would be discussed further in the Section 4.3.
3. The proposed GAN-based framework does not work significantly well on TransR and TransD model. Actually, we find it hard to train TransR with random negative sampling as well. It is easy for TransR to over-fit train dataset since TransR has much more parameters and the consistent non-zero losses brought by our GAN framework worsen this problem by pushing the model too hard. TransD also has the over-fitting problem since TransD distributes a projection vector for each entity and each relation which is not appropriate in terms of generalization. We initial TransR and TransD with the entity and relation embeddings learned from TransE as suggested in (Lin et al. 2015; Ji et al. 2015) but still fail to get a noticeable improvement.

4.2 Triplets Classification

Triplets classification is to classify whether a triplet (h, r, t) holds or not, which is a binary classification problem. It is also used in previous studies to evaluate the embeddings learned by the models.

Table 3: Evaluation results about classification accuracy on triplets classification(%).

Model	FB13			WN11		
	Original	GAN-scratch	GAN-pretrain	Original	GAN-scratch	GAN-prertain
SE (Bordes et al. 2011)	75.2	84.2	**84.7**	53.0	60.1	**63.2**
SME(bilinear) (Bordes et al. 2012)	63.7	77.0	**79.6**	70.0	72.1	**74.3**
SLM (Socher et al. 2013)	85.3	**85.6**	83.8	69.9	72.7	**75.3**
TransE (Bordes et al. 2013)	81.5	83.1	**85.2**	75.9	85.1	**85.4**
TransH (Wang et al. 2014)	83.3	85.0	**86.3**	78.8	85.5	**85.9**
TransR (Lin et al. 2015)	82.5	85.4	**86.6**	85.9	85.2	**86.3**
TransD (Ji et al. 2015)	89.1	85.3	**89.7**	**86.4**	82.6	84.0

The experiments are conducted on the WN11 and FB13 datasets released by (Socher et al. 2013) which already have a negative triplet for each positive triplet in the validation set and test set. The negative triplets are constructed by randomly corrupting the entities of the positive triplets but with the constraint that an entity is chosen as the replacement only if it appears at the same position (head or tail) in the dataset. Thus it is not a trivial task for knowledge representation learning.

For triplets classification, we preset a threshold δ_r for each relation r. If the score of (h, r, t) is below δ_r, then the triplet is classified as positive, otherwise negative. We determine the optimal δ_r by maximizing the classification accuracy on the validation set.

Implementation. We use Adam SGD for optimization. We select the margin γ among $\{0.5, 1.0, 2.0, 4.0\}$, the dimension d of entities and relations among $\{20, 50, 100, 200\}$, learning rate λ for SGD among $\{0.01, 0.001, 0.0001, 0.00005\}$ and the batch size B among $\{128, 512, 1024, 2048, 4096\}$. We search the best configuration according to the accuracy on validation set. For "GAN-scratch", the optimal configurations are $\gamma = 4.0$, $d = 50$, $\lambda = 0.001$, and $B = 1024$ on WN11; $\gamma = 1.0$, $d = 100$, $\lambda = 0.0001$, and $B = 4096$ on FB13. For "GAN-pretrain", the optimal configurations are $\gamma = 4.0$, $d = 50$, $\lambda = 0.0001$, and $B = 512$ on WN11; $\gamma = 1.0$, $d = 100$, $\lambda = 0.00005$, and $B = 1024$ on FB13. For both datasets, wo adopt the L1 as dissimilarity and use the "bern" strategy for deciding whether to replace head or tail when training.

Result. Evaluation results are reported in Table 3. We still directly reprint the best "Filter" experimental results of the original models from the literature, which are referred as "Original" in the Table 3. The results show that

1. Models including SE, SME(bilinear), TransE and TransH are further improved under the "GAN-scratch" and "GAN-pretrain" settings, which demonstrates the effectiveness of our method on simple models. Simple models generalize well to test set, and the consistent non-zero losses brought by the generator in our GAN framework make sure that the models fit the training set sufficiently.
2. For sophisticated models like SLM, TransR and TransD, their improvements brought by our method are different. SLM is improved under the "GAN-scratch" setting on FB13 and WN11 datasets while "GAN-pretrain" does not work well on FB13 dataset. TransR is improved under the "GAN-pretrain" setting on FB13 and WN11 datasets while the "GAN-scratch" only yields a better result than the "Original" mode on the FB13 dataset. TransD is only improved under the "GAN-pretrain" setting on the FB13 dataset. This is due to the complexity and overfitting problem discussed in previous Section 4.1, but still demonstrates the ability of our generator to bring non-zero losses continually.
3. Most of the models trained under the "GAN-pretrain" setting outperform those trained under the "GAN-scratch" setting on both datasets, which is consistent with the result of link prediction task in Section 4.1.

4.3 Discussion on GAN-scratch vs. GAN-pretrain

Both link prediction and triplets classification tasks show that the "GAN-pretrain" setting achieves better performance than the "GAN-scratch" setting under most circumstances. One reason is that the pre-trained models have already converged to some less-optimal state but fail to make further progress because of the zero loss problem. Therefore GAN is able to fine-tune them from a better starting point. Another is that the search space of entities that could bring non-zero loss for the "GAN-pretrain" setting is much smaller than that for the "GAN-scratch" setting. After pre-training, most of the negative triplets have been within the margin. Thus the policy network used by the generator in the "GAN-pretrain" setting might face a more stable environment while the one used in "GAN-scratch" face an environment that changes dynamically. The "GAN-scratch" setting has to gradually narrow down the search space from the whole entity set to a few of entities due to the changing reward given by the discriminator in Eq. (15). "GAN-pretrain" overcome this issue by enabling the policy network in generator directly search the entities within the margin since the reward given by the discriminator is already stable.

4.4 Visualization of Negative Triplets

We visualize some typical negative triplets generated by our generator in Table 4. We conduct the experiment on FB13 dataset since its triplets are more interpretable. The positive triplets from the training set are listed at the first row of each cell in Table 4 and the second row lists out the head and tail negative entities generated by our model. We use generator trained under the "GAN-pretrain" setting. From Table 4 we can see that the negative head entities are basically person entities as we expect. The negative tail entities are also quality and informative since each of them is of the

Table 4: Examples of Negative Triplets from Generator on FB13. In each cell, the first row displays the original triplets and the second row displays the head and tail entity generated by our model.

(*william_preyer*, *ho_yuen_hoe*	*gender*,	*male*) *female*
(*ole_rolvaag*, *vasil_levski*	*religion*,	*lutheranism*) *catholicism*
(*parameshvara*, *johann_fussli*	*nationality*,	*india*) *united_states*
(*peter_scott*, *lester_b_pearson*	*institution*,	*royal_academy*) *eton_college*
(*michel_chartrand*, *louis_jordan*	*profession*,	*politician*) *writer*
(*mark_fidrych*, *noah_rosenzweig*	*cause_of_death*,	*accident*) *cancer*

same entity type as their positive counterpart respectively. As an example, for the triplet (*michel_chartrand, profession, politician*), the negative entities generated by our generator is *louis_jordan* (a person name) for the head and *writer* (also a type of profession) for the tail. This justifies the ability of our generator to generate negative triplets that are more challenging for the discriminator than random sampling, which pushes our discriminator to its limit in embedding entities and relations. Thus, our model can learn the real concept of each entity better since the discriminator is trained to distinguish the positive entities from the high-quality negative entities.

5 Conclusion and Future Work

We proposed to incorporate the GAN training framework for knowledge representation learning. The generator could provide much more quality negative triplets than random sampling while the discriminator thus could be helped to learn better embeddings for entities and relations. We show by extensive experiments that the GAN-based framework could efficiently improve several existing knowledge embedding models and the experimental results show that the idea can generalize well to the state-of-the-art knowledge embedding models.

Future work. We will conduct additional analysis on our GAN-based framework for knowledge representation learning and its generalization to other problems that involve negative sampling. Also, we will experiment with a more complicated architecture of the generator.

6 Acknowledgements

This work was supported by the National Key R&D Program of China under Grant 2016YFB0201900, and the Fundamental Research Funds for the Central Universities under Grant 17LGJC23. This work was also supported by iPIN Inc. Shenzhen, China.

References

Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; and Ives, Z. 2007. Dbpedia: A nucleus for a web of open data. *The semantic web* 722–735.

Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; and Taylor, J. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 1247–1250. AcM.

Bordes, A.; Weston, J.; Collobert, R.; Bengio, Y.; et al. 2011. Learning structured embeddings of knowledge bases. In *AAAI*, volume 6, 6.

Bordes, A.; Glorot, X.; Weston, J.; and Bengio, Y. 2012. Joint learning of words and meaning representations for open-text semantic parsing. In *Artificial Intelligence and Statistics*, 127–135.

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, 2787–2795.

Bordes, A.; Glorot, X.; Weston, J.; and Bengio, Y. 2014. A semantic matching energy function for learning with multi-relational data. *Machine Learning* 94(2):233–259.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, 2672–2680.

He, S.; Liu, K.; Ji, G.; and Zhao, J. 2015. Learning to represent knowledge graphs with gaussian embedding. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 623–632. ACM.

Hermans, A.; Beyer, L.; and Leibe, B. 2017. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*.

Ji, G.; He, S.; Xu, L.; Liu, K.; and Zhao, J. 2015. Knowledge graph embedding via dynamic mapping matrix. In *ACL (1)*, 687–696.

Ji, G.; Liu, K.; He, S.; and Zhao, J. 2016. Knowledge graph completion with adaptive sparse transfer matrix. In *AAAI*, 985–991.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; and Zhu, X. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, 2181–2187.

Miller, G. A. 1995. Wordnet: a lexical database for english. *Communications of the ACM* 38(11):39–41.

Schroff, F.; Kalenichenko, D.; and Philbin, J. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 815–823.

Socher, R.; Chen, D.; Manning, C. D.; and Ng, A. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, 926–934.

Suchanek, F. M.; Kasneci, G.; and Weikum, G. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, 697–706. ACM.

Sutton, R. S.; McAllester, D. A.; Singh, S. P.; and Mansour, Y. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, 1057–1063.

Wang, Z.; Zhang, J.; Feng, J.; and Chen, Z. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, 1112–1119.

Wang, J.; Yu, L.; Zhang, W.; Gong, Y.; Xu, Y.; Wang, B.; Zhang, P.; and Zhang, D. 2017. Irgan: A minimax game for unifying generative and discriminative information retrieval models. *arXiv preprint arXiv:1705.10513*.

Wang, B.; Liu, K.; and Zhao, J. 2017. Conditional generative adversarial networks for commonsense machine comprehension. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 4123–4129.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8(3-4):229–256.

Xiao, H.; Huang, M.; and Zhu, X. 2015. From one point to a manifold: Knowledge graph embedding for precise link prediction. *arXiv preprint arXiv:1512.04792*.

Xie, R.; Liu, Z.; Chua, T.-s.; Luan, H.; and Sun, M. 2016a. Image-embodied knowledge representation learning. *arXiv preprint arXiv:1609.07028*.

Xie, R.; Liu, Z.; Jia, J.; Luan, H.; and Sun, M. 2016b. Representation learning of knowledge graphs with entity descriptions. In *AAAI*, 2659–2665.

Xu, J.; Qiu, X.; Chen, K.; and Huang, X. 2017. Knowledge graph representation with jointly structural and textual encoding. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 1318–1324.

Zhong, H.; Zhang, J.; Wang, Z.; Wan, H.; and Chen, Z. 2015. Aligning knowledge and text embeddings by entity descriptions. In *EMNLP*, 267–272.