Knowledge Base Population via
Relational One-Class Graph Convolutional Networks

Anonymous authors

Abstract

We consider the problem of knowledge base population (KBP) of a coauthor network. Specifically, we consider the link prediction problem as a relational modeling task. We develop a relational extension of graph convolutional networks (GCNs) that exploits recent success inside relational probabilistic models. As a result of this extension, we present a novel knowledge base consisting of knowledge for four major machine learning conferences extracted from the Microsoft Academic Graph. Our empirical evaluation for link prediction on a subset of the extracted knowledge base demonstrates the superiority of relational GCNs over various relational embedding models.

1. Introduction

Statistical Relational Learning (SRL) [Getoor and Taskar, 2007, Raedt et al., 2016] combines the power of probabilistic models to capture uncertainty with logic/relational models to take advantage of the rich domain structure. One of the key successes of these models lie in the task of Knowledge base population, specifically, in link prediction or relation extraction. While successful, most methods make several simplifying assumptions – presence of supervision in the form of labels, closed world assumption, presence of only binary relations and most importantly, the presence of hand-crafted rules.

We go beyond these assumptions and inspired by the recent success of Graph Convolutional Networks (GCNs) [Defferrard et al., 2016, Kipf and Welling, 2017], we develop a new relational extension of GCNs. The proposed method is capable of (1) automatically learning rules from one-class data, i.e., only from the positive annotations of the relation (in our case, coauthor) (2) automatically converting these rules into observed features and (3) training a GCN that is specific to the task of link prediction. Our method employs a one-class density estimation method that employs a tree-based distance metric to learn relational rules iteratively. We call this method as Relational One-Class GCN (ROCGCN). Since the two different steps of learning the relational rules and the GCN training both employ the same set of positive examples, they allow for richer representation of the combi-
nation of the attributes, entities and their relations. While previous methods simply used the observed features as the observed layer, our method uses the combinations (rules) as the observed layer. This has the added advantage of the latent layer being richer – it combines the combinations (rules) themselves allowing for a more richer representation. As we show empirically, this is specifically useful when employed on large-scale KBP tasks such as link prediction.

We make a few key contributions: (1) We develop the first relational GCN capable of learning from positive and unlabeled data. (2) Instead of employing hand-crafted rules, our method learns these rules automatically by focusing on the link prediction task and then constructs the GCN for the same task. These two steps that are conditioned on the link prediction task allows for a better classifier. (3) ROCGCN can handle arbitrary relations – we do not make the assumption of binary relations that most methods make. Given that our base learner is a logic-based one, the predicates can include arbitrary number of parameters. (4) Finally, an important contribution of this work is the release of Microsoft Knowledge Graph in the form of first-order logic factbase. This will allow for benchmarking several SRL models in the immediate future.

The rest of the paper is organized as follows: after briefly reviewing the background, we present our relational GCNs. Then we show how they are used for the link prediction task. Finally, we present empirical results before discussing areas of future research.

2. Background and Related Work

Before we discuss our ROCGCN approach, we present the necessary background by discussing graph embeddings and GCN with their connection to KBP.

2.1 Knowledge Graph Embeddings

Recently, several successful methods for learning embeddings of large knowledge bases have been developed. They have been motivated through the inevitability of learning and reasoning about various entities, their attributes and relations present in the knowledge bases. Several of these approaches such as TransE [Bordes et al., 2013], TransH [Wang et al., 2014], TransG [Xiao et al., 2016] and KG2E [He et al., 2015], to name a few, can be grouped into translational distance models that focus on minimizing a distance based function under some constraints or using regularizing factors between entities and relations. More recent approaches propose to extend these translation approaches by embedding the knowledge graphs into more complex spaces such as the hyperbolic space [Balazevic et al., 2019, Kolivakis et al., 2019] and the hypercomplex space [Zhang et al., 2019, Sun et al., 2019]. Another important class of approaches such as RESCAL [Nickel et al., 2011], DistMult [Yang et al., 2015], TuckER [Balazevic et al., 2019b], HypER [Balazevic et al., 2019a] and HoIE [Nickel et al., 2016] focus on various compositional operators for the entities and relations in the knowledge graph. A more complete survey on the different types of knowledge graph embeddings is given in [Wang et al., 2017] and [Cai et al., 2018].
2.2 Graph Convolutional Networks

While several methods have focused on applying neural networks to extract relational embeddings on large networks such as knowledge graphs, a potential limitation of much of this line of research is that the inherent semantic structure of the network is not exploited. To overcome this limitation, graph convolutional networks (GCNs) [Kipf and Welling, 2017] was proposed that generalized neural network models to multi-relational, graph-structured data sets. A GCN [Kipf and Welling, 2017] \( \mathcal{G} \) generalizes neural network models to multi-relational, graph-structured data sets where each convolution layer in the GCN applies a graph convolution i.e. a spectral filtering of the graph signal (the feature matrix of the graph) via the Graph Fourier Transform.

The basic idea behind GCNs is shown in figure 1. The main reason behind the success of GCNs is that they exploit two key types of information: node feature descriptions \( x_i \) and node neighborhood structure (captured through the adjacency matrix \( \mathcal{A} \) of the graph). Similar to the convolution operators in convolutional neural networks (CNNs) that extract locally stationary features in the inputs data, GCNs utilize the graph convolution operator defined with respect to the adjacency matrix \( \mathcal{A} \) to extract local features from a semantic point of view.

The fundamental difference between CNNs and GCNs is that, while CNNs apply spatial convolutional filters, GCNs apply spectral convolutional filters or similar aggregational filters [Hamilton et al., 2017, Ying et al., 2018] to incorporate the neighborhood information in the underlying model which leads to a better generalization. Similar to a CNN which can, and generally has, multiple convolutional layers, successive application of graph convolution is possible, with each application (layer) producing a successively more informative node embeddings, while possibly simplifying the network structure.

While successful, these GCNs still have a limitation in that they cannot directly be applied on multi-relational data/networks. To alleviate this, a recent extension to the relational data for GCNs was proposed [Schlichtkrull et al., 2018] where a latent representation
of the entities are explicitly constructed and a tensor factorization then exploits these representations for the prediction tasks. We take an alternative approach – we consider the more recent successes inside SRL [Khot et al., 2014, Lao and Cohen, 2010] to develop novel combinations of the entities and their relationships to construct richer latent representations. As we demonstrate empirically, this leads to superior predictive performance.

While our work can be used for entity attribute prediction and link prediction, we focus on the more interesting link prediction task to demonstrate the utility of our approach.

3. Relational Graph Convolution Networks

As mentioned earlier, GCNs by themselves cannot fully exploit the inherent structures inside a multi-relational graph. Motivated by this, we propose an extension to the graph convolutional network that can handle large multi-relational networks. Although a recent work [Schlichtkrull et al., 2018] that extends the GCN framework to relational domains exists, this approach is still limited to graphs represented as (subject; predicate; object) triples. We propose a novel and a more general approach that is not limited by assumptions regarding the multirelationality of the data and can handle general multi-graphs and hypergraphs implicitly and without loss of information. Figure 2 shows a representation of our approach, Relational One Class Graph Convolutional Networks (ROCGCN), to construct the feature matrix \( \mathbf{X} \) containing the node feature descriptors \( x_i \) and the distance matrix \( \mathcal{D} \).

Figure 2: Feature and Distance Matrix Construction for Relational One-Class Graph Convolutional Networks

Before we define the process of obtaining \( \mathcal{D} \) and \( \mathbf{X} \) we present some required background. We first introduce some basic notation and concepts in first-order logic. An atom is of the form \( R(t_1, \ldots, t_k) \) where \( R \) is a functor and the arguments \( t_i \) are terms. A substitution is of the form \( \theta = \{ \langle x_1, \ldots, x_k \rangle / \langle t_1, \ldots, t_k \rangle \} \) where \( x_i \)'s are logical variables and \( t_i \)'s are terms. A grounding of an atom with logical variables \( x_1, \ldots, x_k \) is a substitution \( \{ \langle x_1, \ldots, x_k \rangle / \langle X_1, \ldots, X_k \rangle \} \) mapping each of its variables to a constant in the population of that variable. A knowledge base \( \mathcal{B} \) consists of (1) entities: a finite domain of objects \( \mathcal{D} \), (2) relations: a
set of predicates \( R \) describing the attributes and relationships between objects \( \in D \), and (3) an interpretation assigning a truth value to every grounding.

Given a knowledge base \( B \), we first learn a set of first-order rules that represent the relational search space effectively. The intuition is that these relational rules can be viewed as meta-features that connect entities and their attributes. Particularly, when learned for a specific task, these features can be both predictive and informative. Given that they are typically conjunctions of relational features (attributes of entities and relationships), they have the added advantage of being interpretable. Our hypothesis, that we verify empirically is that these rules can potentially yield richer latent representations than a relational GCN that simply uses the entity and relationship information.

A common issue in many real-world relational knowledge bases is that only true instances of any relation(s) are present while the false instances are not explicitly identified as they are prohibitively expensive. Consequently closed world assumption is applied to sample negative instances from the set of unobserved ones. While successful, this is a strong assumption particularly when the number of positively labeled examples are sparse such as our coauthor network. Inspired by the success of learning only from positive examples in relational domains [Khot et al., 2014], we learn first-order rules using relational density estimation. The intuition behind using a density estimation method is as follows: Learning first order rules for positive and sampled negative examples independently can result in better utilization of the search space thereby learning more discriminative features. The density estimation approach uses a tree-based distance measure that iteratively introduces newer features (as short rules) that covers more positive examples. These rules at the end of the density estimation step form the relational features for our model.

The learned first order rules are then grounded to obtain all instantiations of these rules. The counts of each feature, i.e., the count of the number of times a target example (the coauthor relation between the target entities) is satisfied in every first order rule is obtained which forms our feature matrix \( X \). For example, the learned first order rule from true instances

\[
\text{CoAuthor(person}_1,\text{person}_2) \iff \text{Affiliation(person}_1,\text{university}_1) \\
\land \text{Affiliation(person}_2,\text{university}_1) \land \text{ResearchTopic(person}_1,\text{topic}_1) \\
\land \text{ResearchTopic(person}_2,\text{topic}_1).
\]

implies that if 2 people have the same affiliation and their research interests lie in similar topics then they are more likely to coauthor. Suppose the given target entities are \( \text{person}_1 = \text{“Pieter Abbeel”} \) and \( \text{person}_2 = \text{“Sergey Levine”} \). The partially grounded first order rule can then be written as

\[
\text{CoAuthor(Pieter Abbeel, Sergey Levine) } \iff \text{Affiliation(Pieter Abbeel, university}_1) \\
\land \text{Affiliation(Sergey Levine, university}_1) \land \text{ResearchTopic(Pieter Abbeel, topic}_1) \\
\land \text{ResearchTopic(Sergey Levine, topic}_1).
\]

Then substitutions for all the other entities within the first order rule is performed and checked whether the substituted first order rule is satisfied in the groundings. For example, the substitution \( \theta = \{\langle \text{university}_1, \text{topic}_1 \rangle/\langle \text{University of California Berkeley, Artificial Intelligence} \rangle \} \) is satisfied but the substitution \( \theta = \{\langle \text{university}_1, \text{topic}_1 \rangle/\langle \text{University of California Berkeley, Computer Networks} \rangle \} \) is not satisfied. Since there can be multiple
values taken by $\text{topic}_1$ that can satisfy the first order rule, the count of all such satisfied groundings becomes a feature value for the target query $\text{CoAuthor}(\text{Pieter Abbeel, Sergey Levine})$. Thus we obtain a feature set $\mathcal{X}$ of size $n \times k$ where $n$ is the number of target queries and $k$ is the number of first order rules.

In order to obtain the distance matrix $\mathcal{D}$ a pairwise euclidean distance of all the node feature descriptors i.e. the counts $x_i \in \mathcal{X}$ is computed. Thus, every element $d_{ij} \in \mathcal{D}$,

$$d_{ij} = \|x_i - x_j\| = \sqrt{\|x_i\|^2 + \|x_j\|^2 + 2 \cdot x_i \cdot x_j} \quad (1)$$

The original formulation of graph convolutional networks [Kipf and Welling, 2017] require an adjacency matrix $\mathcal{A}$ to perform the layer wise propagation. Instead of building the adjacency matrix from the relation triples, we use the computed distance matrix $\mathcal{D}$ and use it as an approximation to the adjacency matrix for the GCN. To obtain this approximation, we perform the following steps:

1. A threshold, $t$, is set as the average of all the distances (since the distance matrix is symmetric, the average is calculated from the upper-right part of the distance matrix). This case is considered as far-away case.

2. $\forall d_{ij} \in \mathcal{D}$, new distances are computed as $\hat{d}_{ij} = d_{ij}/t$ and max($\hat{d}_{ij}$) is set as 1.

3. Since the higher values in $\mathcal{D}$ represent nodes that are far way as opposed to the $\mathcal{A}$ where the higher values i.e. 1 represents the nodes adjacent to each other, the distance between nodes is subtracted from 1 i.e. $\hat{d}_{ij} = 1 - \hat{d}_{ij}$. This is similar to $\mathcal{A}$ with $d_{ij} = 1$ representing that two nodes are connected and $d_{ij} = 0$ representing that two nodes are not connected with the only difference being the presence of values $0 < d_{ij} < 1$ that denote the closeness of two nodes.

4. Since the above operator sets the diagonals of the distance matrix $\mathcal{D}$ to 1, in order to approximate the adjacency matrix, the values in the diagonals are reverted to 0.

For a GCN with $M$ layers, the layer wise propagation rule for the layer $l \in M$ can now be written as,

$$f(H^{(l)}, \mathcal{D}) = \sigma(\mathcal{D}H^{(l)}W^{(l)}) \quad (2)$$

where $H(0)$ is the input layer i.e. the feature matrix $\mathcal{X}$ with $H(1) \ldots H(M - 1)$ being the hidden layers. Since we replace $\mathcal{A}$ with $\mathcal{D}$ before the symmetric normalization and addition of self loops, these operations are now be performed on $\mathcal{D}$ thereby giving the updated propagation rule as,

$$f(H^{(l)}, \hat{\mathcal{D}}) = \sigma(\tilde{\mathcal{N}}^{-\frac{1}{2}} \hat{\mathcal{D}} \tilde{\mathcal{N}}^{-\frac{1}{2}} H^{(l)}W^{(l)}) \quad (3)$$

such that $\hat{\mathcal{D}} = \mathcal{D} + \mathcal{I}$ where $\mathcal{I}$ is the identity matrix and $\tilde{\mathcal{N}}$ is the diagonal node degree matrix of $\mathcal{D}$.

Given that we have outlined the procedure for constructing our ROCGCN, we now proceed to discuss the KBP construction using ROCGCN.
4. Knowledge Base Extraction from Microsoft Academic Graph

To create our knowledge base (KB), we extract data from Microsoft Academic Graph (MAG) [Sinha et al., 2015] for 4 major AI/ML conferences, namely International Conference on Machine Learning (ICML), International Conference on Learning Representations (ICLR), International Joint Conference on Artificial Intelligence (IJCAI) and Neural Information Processing Systems (NeurIPS). The data is extracted by querying the academic graph with the keyword ‘venue’. For each conference, we extract title, #citations, year, author(s), affiliation of the author(s), every author’s field of study i.e. the research topic and the citation of each paper. All the obtained information is then converted into a first order logic representation which are collectively referred to as facts. Thus a set of facts $\in B$ describe relational examples through their attributes and relationships. The citation information is then used as the positive examples in the KB. To generate the negative examples for the baselines, the closed world assumption was used. First, all pairs of citation (relations of the form citations($citing\, paper\, id$, $cited\, paper\, id$)) were collected and the cross-product of these combinations were generated. The pairs not in the positive examples constitute the negative examples thereby completing the construction of the KB. The number of facts, positive and negative examples grouped by each conference is shown in the table 1.

<table>
<thead>
<tr>
<th>Conference</th>
<th># facts in Knowledge Base</th>
<th>#Positive examples</th>
<th>#Negative examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICML</td>
<td>71277</td>
<td>77526</td>
<td>142079901</td>
</tr>
<tr>
<td>ICLR</td>
<td>2764</td>
<td>4086</td>
<td>424937</td>
</tr>
<tr>
<td>IJCAI</td>
<td>115575</td>
<td>98730</td>
<td>383165323</td>
</tr>
<tr>
<td>NeurIPS</td>
<td>107368</td>
<td>110926</td>
<td>308649177</td>
</tr>
</tbody>
</table>

Table 1: Properties of the extracted knowledge base grouped by conference

Figure 3 & 4 shows the top 5 fields of study for the papers accepted in the 4 major conferences and the count of the number of authors, unique papers and the coauthor relations in the extracted knowledge base respectively. Not surprisingly, “Machine Learning” is the field of study for most of the papers in all the conferences except IJCAI since these are primarily machine learning conferences. In IJCAI, papers with “Computer Science” as the field of study outnumber the other fields showing that papers with a more general focus are accepted in the conference.

Also papers with the field of study as “Mathematics” appear in every conference thereby depicting the connection of the areas of mathematics and ML/AI. Some more interesting patterns are visible such as the presence of a lot of “Mathematical optimization” papers in NeurIPS and “Algorithms” papers in IJCAI thereby showing the difference in focus of these conferences.

5. Experimental Results

Recall that our original goal is to perform link prediction on the generated KB. To evaluate the algorithm, we consider 3 knowledge bases: ICML’18 consists of papers from ICML 2018, ICLR consists of papers from ICLR (2013-2019) and prediction task is whether two people are coauthors. The third knowledge base DDI is a drug-drug interaction data set [Dhami
et al., 2018] consisting of 78 drugs and prediction task is whether two drugs interact. The various properties of these knowledge bases are shown in table 2.

<table>
<thead>
<tr>
<th>Knowledge Base</th>
<th># facts</th>
<th>#Positive examples</th>
<th>#Negative examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICML’18</td>
<td>1395</td>
<td>155</td>
<td>6498</td>
</tr>
<tr>
<td>ICLR</td>
<td>4730</td>
<td>990</td>
<td>10000</td>
</tr>
<tr>
<td>DDI</td>
<td>1774</td>
<td>2832</td>
<td>3188</td>
</tr>
</tbody>
</table>

Table 2: Properties of various knowledge bases used in the experiments

The reduced KB of 4 authors and their field of study (research topics) is shown in Figure 5. Note that some interesting properties about the entities in the KB can be extracted. For example, “Yann LeCun” and “Pieter Abbeel” have not coauthored and do not share affiliation but the KB suggests that their fields of study highly intersect with one another.

As the first step, we learn the first order logic rules using the density estimation method of [Khot et al., 2014] using relational trees from positive examples. The number of rules learned for both ICML’18 and ICLR is 7 and the for DDI is 25. Some examples of the learned rules for ICML’18 knowledge base are given in table 3. The feature matrix $X$ and the distance matrix $D$ are then obtained following the method described in section 3.


Table 3: Example rules for ICML’18 learned by the density estimation method for +ve and -ve examples. Here “MO” = Mathematical Optimization and “PR” = Pattern Recognition.
5.1 Baselines

We compare our method, ROCGCN, to 6 relational embedding baselines. Gaifman models [Niepert, 2016]: makes use of the Gaifman locality principal [Gaifman, 1982] to simply enumerate all hand-written first order rules (relational features) of the specific kind within the neighborhood of the target/query variables. For our experiments handwritten rules for the reduced knowledge base are constructed. After obtaining the counts for the satisfied grounded handwritten rules, which serve as propositional features, logistic regression is used for prediction. Relational GCN [Schlichtkrull et al., 2018]: extends the GCN to the relational setting. This method can handle different weighted edge types i.e. relations and uses a two step message passing techniques to learn new node representations. For the task of link prediction these node representations are fed to a factorization method, DistMult [Yang et al., 2015], to predict the possible link between the nodes. We use the tensorflow implementation\(^1\) for our baseline. ComplEx [Trouillon et al., 2016]: proposes a latent factorization approach for the problem of link prediction in multi-relational graphs and uses vectors with complex values for entities and relations. The scoring function for each example (a triple) is a combination of the real and imaginary parts of the vectors of both the entities and the relations. We use the ComplEx implementation in the AmpliGraph python library\(^2\) for our baseline. ConvE [Dettmers et al., 2018]: uses convolutions over embeddings and fully connected layers to model interactions between input entities and relationships. We use the ConvE implementation in the AmpliGraph python library for our baseline. SimpleIE [Kazemi and Poole, 2018]: adapts the concept of Canonical Polyadic decomposition and learns two dependent embeddings for each entity and relation to obtain a similarity score for each triple to perform link prediction. We use the tensorflow implementation\(^3\) for our baseline. Node+LinkFeat [Toutanova and Chen, 2015]: is obtained by learning a logistic regression model over the learned propositional features. A comparison of our method, ROCGCN, with various baselines is shown in table 4.

2. https://github.com/Accenture/AmpliGraph
Table 4: Comparison of our method with state of the art baselines

For ROCGCN, the examples for training, validation and testing are randomly sampled without replacement while for neural embedding baselines, since they are trained on true relations, the positive examples are randomly split with 80%, 10% and 10% examples in training, validation and testing respectively. To obtain the different metrics for the neural embedding baseline, the scores for each pair of nodes in the test examples were thresholded by the average of the obtained scores. If the score obtained between a pair of nodes is greater than average score we predict the link else we predict no link. As can be seen from the results, our method outperforms the baselines significantly in 2 of the 3 knowledge bases. Note that although the recall is high for the neural embedding baselines, the corresponding F1 score and AUC-PR is very low which implies that these methods have a high rate of false positives. This clearly demonstrates that ROCGCN is significantly better than the strong baselines for the link prediction task.

6. Discussion

We presented the first GCN method that can learn from positive only multi-relational data. Our ROCGCN does not make assumptions on the supervision or the arity of predicates and automatically constructs rules that allow for a rich latent representation. ROCGCN significantly outperforms the recently successful methods on link prediction task in the knowledge base extracted from the Microsoft Academic Graph. Allowing for joint learning and inference over multiple types of relations is an important future direction. Finally, learning with the presence of hidden/latent data is essential for deploying SRL methods in real tasks.
References


Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. Learning to represent knowledge graphs with gaussian embedding. In CIKM, 2015.


