RA-GCN: Relational Aggregation Graph Convolutional Network for Knowledge Graph Completion

Anqi Tian, Chunhong Zhang, Miao Rang, Xueying Yang and Zhiqiang Zhan
Beijing University of Posts and Telecommunications, Beijing, China

+86-150-0105-8569

kittytian2015@foxmail.com; zhangch@bupt.edu.cn; rmiao2015@163.com;
954179851@qq.com; zqzhan@bupt.edu.cn

ABSTRACT

Knowledge graphs display various entities and their relationships in the real world based on knowledge representation and can analyze and predict the intrinsic relationship between knowledge embeddings through data mining and information processing, which are widely used in search engines, web analytics and smart recommendation areas. As existing knowledge graph information is continuously created and grown, it is a curial task to determine whether the information in knowledge graph is correct and to complete the missing information. In response to this challenge, the researchers proposed a number of graph convolutional network (GCN)-based models to characterize knowledge graphs. The state-of-the-art model is R-GCN, which can effectively extract features. This paper deeply studies the algorithm ideas and results of the R-GCN model, explores whether the model can be further optimized in entity classification and link prediction, and finally, improves the original model and proposes a relational aggregation graph convolutional network. Specifically, this paper finds that a subset of the set of entities may be directly connected to a central entity. All the entities in this subset possess partially identical attributes. At the same time, the relationships between these entities and the central entity may be similar. These similar attributes and relationships can be abstractly aggregated into virtual entities and virtual relationships, respectively, to better extract the topological relationship features. This paper uses the FB15k dataset to evaluate the performance of the proposed model on knowledge graph completion tasks. The experimental results show that the proposed RA-GCN model achieves a certain level improvement compared with the original model and that it can extract knowledge graph topological relationship characteristics more effectively.

CCS Concepts

Computing methodologies → Machine learning → Machine learning approaches → Neural networks

Keywords

Knowledge graph; Knowledge graph completion; Graph

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author.

ICMLC 2020, February 15–17, 2020, Shenzhen, China

© 2020 Copyright is held by the owner/author(s).

ACM ISBN 978-1-4503-7642-6/20/02.

DOI: <https://doi.org/10.1145/3383972.3384067>

convolutional network

1. INTRODUCTION

Any information in a knowledge graph can be represented by a triple (entity 1, relationship, entity 2) or by a connection of several triples. Such data are also called multirelational data. However, knowledge graphs based on multirelational data face many challenging problems. First, graphs with multirelational data are still difficult to understand, and relationships may be missing or invalid. Second, redundancy may exist between entities because several nodes may actually refer to the same concept. Third, most multirelational graphs are very large in terms of the number of entities and the types of relationships they contain, which makes knowledge representation difficult.

To tackle these problems when representing knowledge in multirelational graph data, we reviewed the methods from deep learning and data mining (not covered in detail in this paper)—including node embeddings for statistical relational learning [9], geometric deep learning [2] for non-Euclidean data in social networks and nonlinear learning algorithms [7], etc. In contrast, a graph convolutional network (GCN) [6] is more suitable for extracting the characteristics of large-scale multirelational knowledge graphs.

This paper investigates how to extract the topological relationship features of knowledge graphs based on a GCN, which is based on the spectral domain. A GCN [6] uses a local first-order approximation of spectral convolution to propose a simple and well-performing hierarchical propagation rule suitable for multirelational data. R-GCN [11] is an extension of GCN [6] that extends the hierarchical propagation rules and increases the types of relationships between entities, which makes the model suitable for directed graphs, more in line with general knowledge graph structure and more applicable to the two knowledge graph completion tasks of entity classification and link prediction.

The idea focused on of this paper is based on this recently proposed R-GCN [11] model. The R-GCN [11] algorithm does not consider attribute or relationship similarity between entities. Therefore, we propose an extension, RA-GCN, in this paper. The main contributions of this paper include the following: we improve the propagation formula for updating entity or node information and extract additional entity and relationship information through aggregation. The results show that RA-GCN is better than R-GCN [11] at link prediction and that for entity classification, RA-GCN on the FB15K dataset reaches a classification accuracy 0.13% higher than that of the original model.

2. RELATED WORKS

2.1 GCN Algorithm Principle

In graph convolution theory, in [4], a sample node $X \in R^N$ with a convolutional kernel $g_\theta = \text{diag}(\theta)$ (parameter $\theta \in R^N$ in the Fourier domain), the spectral graph convolution is defined as follows:

$$g_\theta * x = U g_\theta U^T x \quad (1)$$

where U is the eigenvalue matrix of the Laplacian matrix of the graph, the Laplacian matrix of the graph is

$$L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}, \quad I_N \text{ represents the identity matrix, and } D_{ii} = \sum_j A_{ij} \text{ is the definition of the degree matrix.}$$

Intuitively, in Equation (1), x includes neighboring node information because the Laplacian matrix L contains global information about the graph. To improve the calculation speed of

Equation (1), g_θ can be approximated by the Chebyshev polynomial. By setting some parameters, the following layer propagation formula can be generated:

$$Z = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X \Theta \quad (2)$$

In Equation (2), $Z \in R^{N \times F}$ is the convolved output, $X \in R^{N \times C}$ and has C -dimensional feature vectors for every node, $\Theta \in R^{C \times F}$ is the matrix of kernel parameters, and F indicates the number of features after dimension reduction in the output layer because the key point of the graph convolution operation is to reduce the sample dimensions.

Kipf and Welling's [6] GCN model is based on the abovementioned graph convolution principle. They calculated the approximate decomposition of the local convolution, which has a rank of 1. GCN uses the following notations to represent an undirected graph: $G = (V, E)$ with N nodes and $v_i \in V$, edges, where $(v_i, v_j) \in E$. Equivalently, the notation

$G = (A, X)$ with an adjacency matrix $A \in R^{N \times N}$, $X \in R^{N \times F}$ is a matrix of node feature vectors X_i that represent undirected graphs in GCN.

They proposed the node information update in Equation (3), which can obtain the transformation for each layer:

$$h_i^{(l+1)} = \sigma(h_i^{(l)} W_0^{(l)} + \sum_{j \in N_i} h_j^{(l)} W_1^{(l)}) \quad (3)$$

where $h_i \in R^{d^{(l)}}$ are the hidden layer activations of node v_i in the l -th layer of the graph convolutional neural network, and $d^{(l)}$ is the dimension of this layer. In this model, the activation

function $\sigma(\cdot)$ is the $\text{ReLU}(\cdot)$ function. $W_i^{(l)}$ is i -th layer weight matrix. The neural network weights are trained using the gradient descent method. N_i represents the indices of the neighboring nodes of the update node.

This propagation formula has been shown to be very effective in aggregating and extracting structural features from the locality in knowledge graphs and has achieved significant results in semisupervised learning [8] and graphical classification [3]. This type of transformation was improved in R-GCN [11], making the model suitable for large-scale multirelational graph-structured data.

2.2 R-GCN Algorithm Principle

The main enhancement of R-GCN [11], which is based on GCN [6], was to expand the model from a practical undirected graph to a practical directed graph by adding edge relationship types. Compared with the original GCN model, the relations are directed, but the adjacent matrices used for representation contain both each relation and its reverse. To solve this problem, a weighed sum of neighbors from different relations was added to the original model and regularization was applied to the weights to reduce the overfitting problem.

R-GCN [11] defines the following notation to represent knowledge graphs: $G = (V, E, R)$, which represents a directed graph with labels containing N nodes (entities) $v_i \in V$, edges (relations) with labels $(v_i, r, v_j) \in E$, where $r \in R$ is the relation type on an edge and R is the set of relations. R-GCN [11] redefines the propagation rule for calculating the entity (node) update as follows:

$$h_i^{(l+1)} = \sigma(h_i^{(l)} W_0^{(l)} + \sum_{j \in N_i^r} \sum_{r \in R} h_j^{(l)} W_r^{(l)}) \quad (4)$$

where $W_r^{(l)}$ is the relation weight matrix, and different relation types have different weight matrices, $h_i^l \in R^{d^{(l)}}$ is the hidden state of node (entity) v_i in the l -th layer of the neural network, and $d^{(l)}$ is the dimensionality of the layer's representations. The symbol $\sigma(\cdot)$ denotes a nonlinear activation function (such as $\text{ReLU}(\cdot)$), and N_i^r denotes the set of neighbor indices of node i under relation $r \in R$. $W_r^{(l)}$ and $W_0^{(l)}$ are defined as follows:

$$W_r^{(l)} = \sum_{b=1}^B a_{rb}^{(l)} V_b^{(l)} \quad (5)$$

which is a linear combination of the basis transformation $V_b^{(l)} \in R^{d^{(l+1)} \times d^{(l)}}$ with the coefficients $a_{rb}^{(l)}$. Notably, both the coefficients $a_{rb}^{(l)}$ and the transformation $V_b^{(l)}$ are trainable model parameters that are shared over all locations in the graph.

2.3 Limitations of R-GCN and the Improvement Ideas of this Paper

Based on the algorithm principle of R-GCN, this model does not consider the similarity between nodes and relationships that have the same attribute information. In view of the applications of graph convolutional networks, the essence of knowledge graphs is to extract the feature information of the nodes and relationships in

the graph and use that information to update the entire graph layer by layer. To more comprehensively extract the feature information from nodes and relations, this paper proposes the RA-GCN (relational aggregation GCN) and improves the propagation formula. In other words, a collection of entities directly connected to a central entity may have a subset in which all entities have partially identical attributes, while at the same time relationships exist between these entities and the central entity. These may also have similarities; thus, the identical attributes and similar relationships can be abstractly aggregated into virtual entities and virtual relationships.

3. RELATIONAL AGGREGATION GRAPH CONVOLUTIONAL NETWORK

Based on the R-GCN propagation formula in Equation (4), this paper proposes the improved propagation formula shown in Equation (6):

$$h_i^{(l+1)} = \sigma(h_i^{(l)}W_0^{(l)} + \sum_{j \in N_i^r} \sum_{r \in R} h_j^{(l)}W_r^{(l)} + \sum_{a \in A} \frac{1}{c_a} h_a^{(l)}W_a^{(l)}) \quad (6)$$

which calculates an entity or node represented by v_i in the multirelational graphs and is a forward propagation algorithm—that is, a layer-by-layer iteration from the input layer to the output layer. Thus, it traverses the information of all entities or nodes.

The difference between Equation (6) and Equation (4) is that this

paper adds the relational aggregation term $\sum_{a \in A} \frac{1}{c_a} h_a^{(l)}W_a^{(l)}$, in

which a corner a represents the relationship aggregation, A represents the set of virtual entities and virtual relationships after relationship aggregation, and $h_a^{(l)}$ represents the hidden activations of the layer l virtual entity. $W_a^{(l)}$ represents the weight of the relationship aggregation in layer l , which can be

obtained by training the model, and $\frac{1}{c_a}$ is the coefficient of the

item, which is a constant term that can be preset by humans or obtained by training the model.

In Figure 1, the central entity to be updated is shown in pink, and the nodes adjacent to it are blue. First, the process finds entities with the same attributes (for example: people) and abstracts them into a virtual entity, represented by the dotted entity nodes in the figure. Simultaneously, the relationships between these entities with the same attributes and the central entity may be similar. In other words, from these neighboring nodes (blue) with the same attributes, we can collect the relationship vectors associated with the entity to be updated (pink) and aggregate the relationships with similar relationship types

(e.g., "/people/person/spouse(s)/people/marriage/type_of_union" and "/people/person/employment_history./business/employment_tenure/title" belong to the "/people/person/" relationship type and are considered similar). These relationships form an aggregate virtual relationship represented by the dashed relationships in the figure to transform each type of relationship (for incoming and outgoing edges) such as this. Finally, the

obtained layer vector representation is passed to the next layer through the activation function (which is $\text{ReLU}(\cdot)$ in RA-GCN, the same as in R-GCN). The model performs updates iteratively layer by layer to complete the information update process for all entities or nodes.

Figure 1. Diagram for updating a single node in RA-GCN.

4. EXPERIMENTS

4.1 Entity Classification

In knowledge graphs, many node labels are incomplete (e.g., Country is the label of USA). The entity classification task aims to predict missing node (entity) labels (attributes).

Apply the RA-GCN model to the entity classification approach as shown in Figure 2.

Figure 2. The structure of the RA-GCN model on entity classification.

In this structure, based on the original GCN model, the $\text{Softmax}(\cdot)$ function is used to calculate the cross-entropy loss; then, the node loss in the output layer can be obtained.

Based on the original GCN model, Equation (7) shows the architecture of this model:

$$Z = \text{softmax}(\hat{A} \text{ReLU}(\hat{A}XW^{(0)})W^{(1)}) \quad (7)$$

where $\hat{A} = A + I_N$ denotes the preprocessing of the adjacency matrix A , and I_N is the identity matrix, which can be represented as $I_N = \text{diag}(1, 1, \dots, 1)$. $W^{(0)}$ is the weight matrix of the input layer to the hidden layer, and $W^{(1)}$ is the weight matrix of the hidden layer to the output layer.

Then, the *Softmax*($\cdot$) classifiers are applied to every node in the graph after the RA-GCN encoder method completes. This layer takes the knowledge representations from the input layer and predicts the missing labels. The parameters of the RA-GCN model are learned by optimizing for the cross-entropy loss as follows:

$$L = - \sum_{l \in y_L} \sum_{f=1}^F Y_{lf} \ln Z_{lf} \quad (8)$$

where y_L is the set of labeled graph node indices, F is the feature map in the output layer, Y is the label matrix and Z is the last output after the *Softmax*($\cdot$) classifier.

4.2 Link Prediction

Link prediction is intended to predict missing relations in new triples (subject, relation, object). Here, $G = (V, E, R)$ represents a directed, labeled knowledge graph. For the training triples, an incomplete subset $\hat{E}$ is chosen rather than the full set E containing all edges. Specifically, this task assigns a triple score function $f(s, r, o)$ to possible edges (s, r, o) to discover whether the possible edges belong to E in $G = (V, E, R)$.

The following Figure 3 shows the structure of the improved model on the link prediction task.

Figure 3. The structure of the RA-GCN model on link prediction.

In this structure, RA-GCN maps each entity $v_i \in V$ to a real-valued vector, $e_i = h_i \in R^{d^{(l)}}$, where h_i^l is defined in Equation (4). The edges of the graph are reconstructed depending on the node representations. In other words, triples (subject, relation, object) are scored by the score function. We regard this

process as a map: $R^d \times R \times R^d \rightarrow R$. The triple (s, r, o) is scored as by follows

$$f(s, r, o) = e_s^T R_r e_o \quad (9)$$

The RA-GCN model learns a vector for each of the entities and relationships e by continuously optimizing the objective function, which is minimized by the stochastic gradient descent method. During the learning process, to enhance the distinguishing ability of the knowledge representation, the following optimization objective function is applied, also called the loss function, which is defined in DistMult [13] as follows:

$$L = - \frac{1}{(1+w)|\hat{E}|} \sum_{(s,r,o,y) \in \Gamma} y \log \sigma(f(s,r,o)) + (1-y) \log(1 - \sigma(f(s,r,o))) \quad (10)$$

In Equation (6) Γ represents the total set of real and corrupted triples, $\hat{E}$ is the incomplete subset of edges, w is a negative set, and $y = 1$ for positive triples and $y = 0$ for negative triples.

The RA-GCN model is trained by negative sampling. For each example to be observed in this task, w negative items are sampled. Random sampling is used to corrupt the subject entity or the object entity of each positive triple. The goal is to optimize the cross-entropy loss function in (10), to allow the model to score higher on the observable triples than on the negative triples during training.

5. RESULTS AND DISCUSSION

5.1 Results and Analysis of Link Prediction

In this link prediction task experiment, we used FB15k [1] as the primary training and test set and FB15k-237 [5] (a subset of FB15k [1]) as a comparison dataset to assess the effect of the RA-GCN model when the number of entities and relations is small. The specific parameters of the above datasets are shown in Table 1.

Table 1. Specific parameters of the datasets

Dataset	Entities	Relations	Train edges	Validation edges	Test edges
FB15K	14,951	1,345	483,142	50,000	59,071
FB15K-237	14,541	237	272,115	17,535	20,466

In this Link Prediction task, the embeddings dimension is 200. We add edge dropout at a dropout rate of 0.2 for the self-loop and 0.4 for the other edges. We apply the Adam optimizer [12] and set the learning rate to 0.01. For the baselines and RA-GCN model, full-batch optimization was performed in the training process.

Because RA-GCN is an improved variant of R-GCN [11], we adopt R-GCN as its natural baseline. Furthermore, we compare the RA-GCN model to two other common link prediction algorithms: HolE [10] and ComplEx [12]. This paper uses their experimental results from [10] and [12] and two commonly evaluated metrics for the results. One is mean reciprocal rank (MRR), and the other is Hits at n (H@n).

In the experiment, the FB15k dataset is used to perform 200 epochs of training iterations for the proposed model RA-GCN and

the original model R-GCN. The training accuracy and training loss values are shown in Figures 4 and 5, respectively:

Figure 4. Training accuracy comparison for RA-GCN and R-GCN [13] on the FB15k [1] dataset.

Figure 5. Training loss comparison for RA-GCN and R-GCN [13] on the FB15k [1] dataset.

As shown in Figure 4, as the number of training sessions increases, the training accuracy of both RA-GCN and R-GCN increases. During training, the initial accuracy of R-GCN is 0.17, the initial value of RA-GCN is 0.14, and the original accuracy is 0.03 percentage points. The accuracy of both methods increases rapidly over the first 20 training epochs and then continued increasing more gradually. After 150 training sessions, the R-GCN approached 0.85, while the RA-GCN approached 0.90—0.05% higher than the original model. The achieved accuracy of RA-GCN in this paper is higher than that of R-GCN.

As shown in Figure 5, as the number of training sessions increases, the training losses of RA-GCN and R-GCN decrease. During training, the initial training loss of R-GCN is 1.79, while that of RA-GCN is 1.77. The initial loss values are similar. The difference between the loss values of the two is 50 times before

training, and the loss value of RA-GCN is smaller than that of R-GCN. After 200 epochs, the RA-GCN loss value approached 1.10, while that of the R-GCN approached 1.20, 0.1 percentage point below the original model loss value. The fact that the improved RA-GCN loss value is smaller than that of R-GCN indicates that the information in the features extracted from the entities and relationships by the aggregation algorithm remain more stable during the training process, which helps ensure the accuracy of relationship prediction.

From Tables 2 and 3, the test method for filtering out the correct triples (Filter) is better than the unconstrained test method (Raw), and the average ranking of the Filter method is lower than that of the Raw method; however, the Hits@10 of the former is higher than that of the latter because the Filter test method filters out the correct triples when ranking. Thus, the Filter method ranks higher and is more accurate than the Raw test method.

Table 2. Results on the FB15k [1] dataset

Model	MRR		Hits@10	
	Raw	Filtered	Raw	Filtered
HoIE [10]	0.248	0.634	0.718	0.814
ComplEx [12]	0.242	0.692	0.759	0.840
R-GCN [11]	0.251	0.651	0.736	0.825
RA-GCN	0.262	0.696	0.760	0.842

Table 3. Results on the FB15k-237 [5] dataset

Model	MRR		Hits@10	
	Raw	Filtered	Raw	Filtered
HoIE [10]	0.124	0.222	0.253	0.391
ComplEx [12]	0.109	0.201	0.213	0.388
R-GCN [11]	0.158	0.248	0.258	0.414
RA-GCN	0.156	0.249	0.264	0.417

As shown in Table 2 in the experiment on the FB15k dataset, the MRR and Hits@10 indicators of RA-GCN are superior to those of the other models. The MRR indicator shows that—compared with R-GCN [11]—RA-GCN performs better by approximately 1.1 percentage points on the RAW index and by approximately 4.5 percentage points higher on the Filtered index. Correspondingly, on Hits@10, for the top three triples, RA-GCN achieved an increase of 3.6 and 2.7 percentage points on Raw and Filtered, respectively.

As shown in Table 3, compared with R-GCN [11], the MRR and Hits@10 indicators of RA-GCN are not highly similar, and both are superior to the HoIE [10] and ComplEx [12] models. One possible reason for this result is that the FB15k-237 [5] dataset is a subset of FB15k [1]; that is, it contains less characteristic information of entities and relationships, and it shows that the improved model RA-GCN proposed in this paper performs better on larger dataset sizes. The effect of the larger dataset is obvious, which also agrees with the idea underlying our model improvements. The larger the amount of entity attribute and relationship feature information obtained by the aggregate extraction is, the more accurate the prediction will be.

5.2 Results and Analysis on Entity Classification

In the entity classification experiment, we still use FB15k [1] for the main training and test sets, and we select the subset of FB15k-237 [5] as the comparison dataset to compare the effect of the RA-GCN model when the volume of physical and relational data is small. The embeddings dimension is 200. In contrast with link prediction, we do not use dropout during entity classification. We set the learning rate to 0.01. The baselines and the RA-GCN model still use full batch optimization and 50 iterations—namely, 50 epochs. For the hidden layers, we use 16 hidden units to avoid memory leakage during model training. Similar to the link prediction experiment, we chose R-GCN [11] as the baseline comparison model. For this task, we adopt accuracy as an evaluation metric.

As shown in Figures 6 and 7, R-GCN [11] and the RA-GCN model are used to perform entity classification experiments on the dataset. As the number of model layers increases, the classification accuracy changes as follows:

Figure 6. Accuracy result on FB15k-237 [5] dataset.

Figure 7. Accuracy result on FB15k [1] dataset.

As Figures 6 and 7 show, RA-GCN and R-GCN [11] achieve the highest accuracy on the entity classification task. On the FB15k-237 [5] dataset, the accuracy of the two classifications for the entities is similar, and neither are above 90%. With 2 model layers, RA-GCN performs better than R-GCN by 4 percentage points.

The entity classification accuracy is generally better on the FB15k [1] dataset than on the FB15k-237 [11] dataset. RA-GCN achieves the highest accuracy with 2-model layers, outperforming the R-GCN by 5 percentage points. This result indicates that the RA-GCN model performs better and more accurately for entity classification when sufficient entity and relationship information is available.

Therefore, we give RA-GCN and R-GCN [1] both 2-layer models execute both models and calculate the average from 10 experimental runs. The results are shown in Table 4:

Table 4. Entity classification results: the average accuracy of 2-layer models

Model	FB15k	FB15k-237
R-GCN	88.88	88.33
RA-GCN	89.01	88.38

As seen from the above table, RA-GCN is superior to R-GCN in both datasets [1]. On the FB15k dataset, the classification accuracy of RA-GCN is 0.13% higher than that of the original model. On the FB15k-237 dataset, RA-GCN is 0.05% more accurate than is the original model. The improved model RA-GCN proposed in this paper is more effective when the dataset is larger, which is also in line with our idea for improving the model. The more entity attributes and relationship features obtained by the aggregation extraction, the more accurate the classification of the entity attributes becomes. The results also show that the model is still insufficient. When the number of model layers is too large, the accuracy rate decreases, and there is no good universality. This problem can be addressed in future work.

6. CONCLUSION

This paper proposed a relational aggregation graph convolutional network (RA-GCN) and demonstrated its effectiveness in two common knowledge graph completion tasks. The results of experiments indicated that the aggregation process can extract additional entity and relationship information. On the link prediction task, RA-GCN performs better than does R-GCN, while on the entity classification task, the classification accuracy of R-GCN on the FB15K dataset is 0.13% higher than that of the original model.

7. REFERENCES

- [1] Bhagat, S., Cormode, G., and Muthukrishnan, S. 2011. Node classification in social networks. *Comput. Sc.* 16, 3 (Mar. 2011), 115-148. DOI= https://doi.org/10.1007/978-1-4419-8462-3_5.
- [2] Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. 2017. Geometric deep learning: going beyond Euclidean data. *IEEE Signal Proc. Mag.* 34, 4 (Jul. 2017), 18-42. DOI= <http://doi.acm.org/10.1109/MSP.2017.2693418>.
- [3] Duvenaud, D., Maclaurin, D., Aguilera-Iparraguirre, J., Gmez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Proceedings of the 28th International Conference on Neural Information Processing Systems* (Montreal, Canada, December 07-12, 2015). MIT Press, Cambridge, MA, USA, 2224-2232.

- [4] Hammond, D. K., Vandergheynst, P., and Gribonval, R. 2011. Wavelets on graphs via spectral graph theory. *Appl. Comput. Harmon. A.* 30, 2 (Mar. 2011), 129-150. DOI= <https://doi.org/10.1016/j.acha.2010.04.005>.
- [5] Kingma, D., and Ba, J. 2014. Adam: a method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)* (San Diego, USA, May 07-09, 2015). arXiv preprint arXiv:1412.6980, Ithaca, NY.
- [6] Kipf, T., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. In *ICLR 2016* (Caribe Hilton, San Juan, Puerto Rico, September 09, 2016). arXiv preprint arXiv:1609.02907.
- [7] Lee, J. A., and Verleysen, M. 2007. *Nonlinear Dimensionality Reduction*. Springer Science and Business Media, New York, NY.
- [8] Liao, W., Xia, J., Du, P., and Philips, W. 2015. Semi-supervised graph fusion of hyperspectral and lidar data for classification. In *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)* (Milan, Italy, July 26-31, 2015). IEEE, Milan, Italy, 53-56. DOI= <http://doi.acm.org/10.1109/IGARSS.2015.7325695>.
- [9] Nickel, M., Murphy, K., Tresp, V., and Gabrilovich, E. 2016. A review of relational machine learning for knowledge graphs. *Proc. IEEE*. 104, 1 (Dec. 2015), 11-33. DOI= <https://doi.org/10.1109/JPROC.2015.2483592>.
- [10] Nickel, M., Tresp, V., and Kriegel, H.-P. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning* (Washington, DC, USA, June 28 – July 02). Omnipress, New York, NY, 809-816.
- [11] Schlichtkrull, M., Kipf, T. N., Bloem, P., Van den Berg, R., Titov, I., and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference* (Heraklion, Greece, June 03 – 07, 2018). Springer, Cham, 593-607. DOI= https://doi.org/10.1007/978-3-319-93417-4_38.
- [12] Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., and Bouchard, G. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on Machine Learning* (New York, NY, USA, June 19 – 24, 2016). JMLR.org, 2071-2080.
- [13] Yang, B., Yih, W.-T., He, X., Gao, J., and Deng, L. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*. (Dec. 2014).