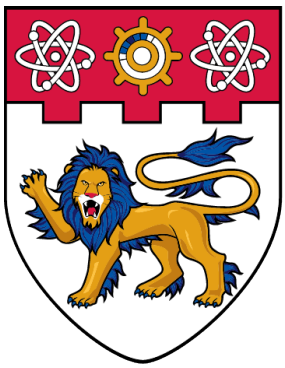

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

CZ4071 Network Science

Project 2: Survey & Implementation of Research Paper

Modeling Relational Data with Graph Convolutional Networks

Michael Schlichtkrull Thomas N. Kipf Peter Bloem
Rianne van den Berg Ivan Titov Max Welling.

Written By:
Haziq Yusoff - U1721504J
Leon Koh Ching Kian - U1721037B
Benedict Low - U1821762E
Zachary Chua - U1721622J
Date: 29 April 2021

Contents

Section 1: Introduction

This report conducts a survey on research paper ”Modeling Relational Data with Graph Convolutional Networks” by Michael Schlichtkrull et alia [1]. It covers the problems investigated by the research paper and its associated applications and challenges. A cross comparison with CZ4071 techniques and problems will also be conducted to determine if there could be possible improvements made to CZ4071 techniques, or if any CZ4071 problems can be solved more efficiently using the content of the research paper.

Section 2: Survey

2.1 Problems Investigated

The two main problems investigated by the research paper are the recovery of missing facts via Link Prediction, and the recovery of missing entity attributes via Entity Classification. The paper specifically targets knowledge bases, which contain and organize factual knowledge.

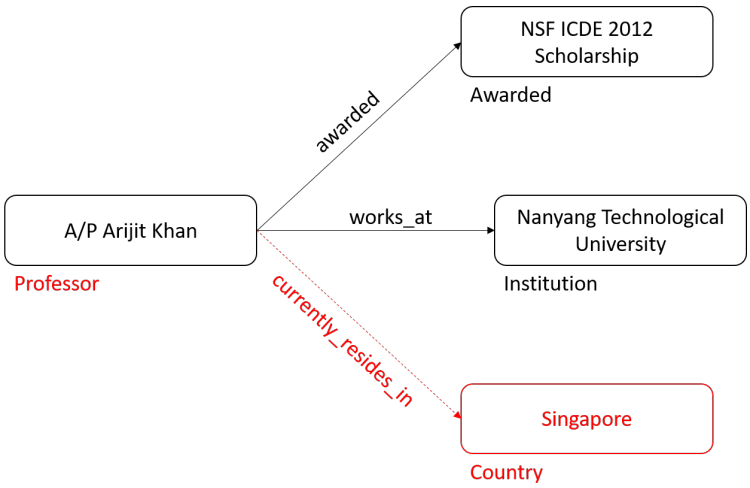

Figure 1: Knowledge Base Fragment Example

Consider Figure 1, a knowledge base fragment with nodes as entities with entity type and edges as relations labeled with relation type. The node and edge labels in red are the missing information to be found. Knowing that A/P Arijit Khan currently works at Nanyang Technological University, implies that:

- A/P Arijit Khan must have the label Professor. **(Entity Classification)**
- The triple (A/P Arijit Khan, currently_resides_in, Singapore) must belong to the knowledge graph. **(Link Prediction)**

Following this theory, an encoder model was developed for entities in the relational graph and applied to the two main problems.

2.1.1 Main Contributions

The main contributions to the paper to tackle the above-mentioned problems are as follows:

1. Prove that GCN Framework can be applied to modeling relational data, specifically for link prediction and entity classification tasks **(detailed in section 2.2.1 and 2.2.2)**.
2. Introduce techniques for parameter sharing and to enforce sparsity constraints to be used to apply R-CGNs to multigraphs with large numbers of relations **(detailed in section 2.2.3)**.

3. Show that the performance of factorization models can be improved via enrichment with an encoder model that performs multiple steps of information propagation in the relational graph. **(detailed in section 2.4)**.

2.2 Algorithms & Machine Learning Techniques Developed

2.2.1 Neural Relational Modelling

The research paper utilizes a modified version of a simple differentiable message-passing framework [2] outlined by Gilmer et al., defined as a simple propagation model for calculating the forward-pass update of an entity or node in a relational (directed and labeled) multi-graph:

$$h_i^{l+1} = \sigma\left(\sum_{r \epsilon R} \sum_{j \epsilon N_i^r} \frac{1}{c_{i,r}} W_r^{(l)} h_j^{(l)} + W_0^{(l)} h_i^{(l)}\right) \tag{1}$$

Where directed & labeled multi-graphs are denoted as $G = (V, \varepsilon, R)$ with nodes (entities) $v_i \epsilon V$ and labeled eges (relations) $(v_i, r, v_j) \epsilon \varepsilon$ where $r \epsilon R$ is the relation type. $h_i^{(l)} \epsilon R^{d^{(l)}}$ is the hidden state of node v_i in the l^{th} later of the neural network of dimensionality $d^{(l)}$. Incoming messages are accumulated and passed through an element-wise activation function $\sigma(\cdot)$ such as $ReLU(\cdot)$. N_i^r denotes the set of neighbour indices of node i under relation $r \epsilon R$.

Equation (1) accumalates transformed feature vectors of neighbouring nodes through a normalized sum. The transformation is primarily used for its efficacy in accumulating and encoding features from local, structured neighbourhoods. It significantly improves graph classification and graph-based semi-supervised learning [3]. Neural network updates consist of evaluating (1) in parallel for every node in the graph. This graph encoder model is referred to as a relational graph convolutional network (R-GCN). This R-GCN generates learned embeddings like in Appendix A that can improve accuracy on downstream ML tasks and is similar to the Word2Vec preprocessing step of NLP applications.

2.2.2 Entity Classification & Link Prediction

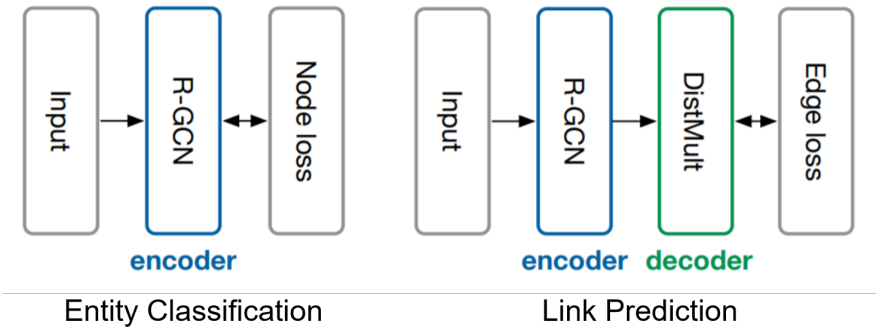

Figure 2: Entity Classification & Link Prediction Model

Figure 2 depicts a schematic for the entity classification and link prediction model.

For semi-supervised classification of nodes (entities), R-GCN layers of the form (1) are stacked with a $softmax(\cdot)$ activation on the output of the last layer. Cross-entropy loss is then minimized on all labeled nodes, ignoring unlabeled nodes:

$$L = -\sum_{i \epsilon Y} \sum_{k=1}^{K} t_{ik} ln\ h_{ik}^{(L)} \tag{2}$$

Where Y is the set of node indices that have labels, $h_{ik}^{(L)}$ is the k^{th} entry of the network output for the i^{th} labeled node, and t_{ik} is the respective ground truth label

Link prediction accounts for the prediction of new facts, or triples (subject, relation, object). However, for labeled graph $G = (V, \varepsilon, R)$ contains an incomplete subset ε. The main task is to assign scores to possible links (s, r, o) to determine the likeliness of the edge belonging to ε.

This task is achieved via a graph auto-encoder model comprising of an entity encoder and a scoring function (decoder), as outlined in Figure 2. The encoder maps every entity $v_i \, \epsilon V$ to a real valued vector $e_i \, \epsilon \, R^d$. The decoder reconstructs edges of the graph via scoring (subject, relation, object) triples. A DistMult factorization [4] was used as the scoring function as it performs well on standard link prediction benchmarks when used on its own. Every relation r is associated with a diagonal matrix $R_r \, \epsilon \, \mathbb{R}^{dxd}$ and a triple (s, r, o) is scored as:

$$f(s, r, o) = e_s^T R_r e_o \tag{3}$$

As with previous works on factorization [4] [5], negative sampling was used to train the model. For each observed example, w negative samples were taken by randomly corrupting either the subject or object of the positive triple sample. Optimization for cross-entropy loss was used to improve the model to score observable triples higher than negative ones:

$$L = -\frac{1}{(1+w)\,|\widehat{\varepsilon}|} \sum_{(s,r,o,y)\epsilon\tau} y \; log \; l(f(s,r,o)) + (1-y) log(1 - l(f(s,r,o))) \tag{4}$$

Where τ is the total set of real & corrupted triples, l is the logistic sigmoid function, and y is an indicator of positive or negative triples, where $y = 1$ indicates a positive triple and $y = 0$ indicates a negative triple.

2.2.3 Regularization

One issue encountered when dealing with highly multi-relational data is the rapid growth in the number of parameters with the number of relations in the graph. This would lead to overfitting on underrepresented relations as well as extremely large model sizes. To combat this, regularization of weights were conducted via two methods, *basis-decomposition* and *block-diagonal-composition*.

Basis-Decomposition:

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

Block-Diagonal-Decomposition:

$$W_r^{(l)} = \bigoplus_{b=1}^{B} Q_{br}^{(l)} \tag{6}$$

Basis-decomposition (5) conducts effective weight sharing between different relation types, alleviating overfitting on underrepresented relations since paremeter updates are now shared between both underrepresented relations and more frequent relations. Block-diagonal-decomposition (6) imposes a sparsity constraint on the weight matrices for each relation type that allows the model being trained to group latent features into sets of variables that are more tightly coupled within groups vs. across groups. Both (5) and (6) also reduce the number of parameters required to learn for highly multi-relational data such as that of in knowledge bases. This is similar to weight sharing in a CNN but instead of a window or filter across a matrix, it shares weights across relations or neighbours.

2.3 How Challenging the Problem Is

There are a few challenges we identified relating to the run-time of the algorithm and the poor performance of R-GCN on data set with inverse triplet pairs.

2.3.1 Time and space complexity

Training enumerates through all n nodes and this would mean a time complexity of O(n). What this means is that the running time is in linear relation to the size of input. This usually is a desired time complexity we want to obtain. However, the goal of using Relational Graph Convolutional Networks (R-GCNs) and applying them to the two standard knowledge base completion tasks could prove challenging. Predicting missing information in knowledge bases such as DBPedia, Wikidata or Yago with extremely large number of triples can take a very long time. A graph with m uncertain components generates 2^m possible worlds.

	DBpedia	Freebase	OpenCyc	Wikidata	YAGO
Number of triples	411 885 960	3 124 791 156	2 412 520	748 530 833	1 001 461 792
Number of classes	736	53 092	116 822	302 280	569 751
Number of relations	2819	70 902	18 028	1874	106
No. of unique predicates	60 231	784 977	165	4839	88 736
Number of entities	4 298 433	49 947 799	41 029	18 697 897	5 130 031
Number of instances	20 764 283	115 880 761	242 383	142 213 806	12 291 250
Avg. number of entities per class	5840.3	940.8	0.35	61.9	9.0
No. of unique subjects	31 391 413	125 144 313	261 097	142 278 154	331 806 927
No. of unique non-literals in object position	83 284 634	189 466 866	423 432	101 745 685	17 438 196
No. of unique literals in object position	161 398 382	1 782 723 759	1 081 818	308 144 682	682 313 508

Figure 3: Key Statistics of Knowledge Graphs

As we can see in Figure 3, the number of triples in the largest knowledge graph can go up to billions and this can prove tough for even the best hardware available.

2.3.2 Data set cleaning

The presence of inverse triplet pairs $t = (e_1, r, e_2)$ and $t_0 = (e_2, r^{-1}, e_1)$ with t in the training set and t_0 in the test set proved to be a challenge in highlighting the need for R-GCN. This reduces a large part of the prediction task to memorization of affected triplet pairs. A simple baseline LinkFeat employing a linear classifier on top of sparse feature vectors of observed training relations was shown to outperform R-GCN by a large margin. In order for R-GCN to achieve its potential and perform better than the baseline LinkFeat algorithm, data cleaning must be done to remove all such inverse triplet pairs.

2.4 Performance

Entity Classification

Model	AIFB	MUTAG	BGS	AM
Feat	55.55	77.94	72.41	66.66
WL	80.55	**80.88**	86.20	87.37
RDF2Vec	88.88	67.20	**87.24**	88.33
R-GCN	**95.83**	73.23	83.10	**89.29**

Link Prediction

	MRR		Hits @		
Model	Raw	Filtered	1	3	10
LinkFeat		0.063			0.079
DistMult	0.100	0.191	0.106	0.207	0.376
R-GCN	**0.158**	0.248	**0.153**	0.258	0.414
R-GCN+	0.156	**0.249**	0.151	**0.264**	**0.417**
CP	0.080	0.182	0.101	0.197	0.357
TransE	0.144	0.233	0.147	0.263	0.398
HolE	0.124	0.222	0.133	0.253	0.391
ComplEx	0.109	0.201	0.112	0.213	0.388

Figure 4: Consolidated Performance for Entity Classification (left) & Link Prediction (right)

For entity classification, the R-GCN model proposed by Schlichtkrull et al. outperformed existing models in 2 of the 4 datasets (AIFB and AM), while obtaining the 3rd highest accuracy for the MUTAG and BGS datasets. For link prediction, the R-GCN and R-GCN+ models (a combination of R-GCN and DistMult) performed the best across all metrics for the FB15k-237 dataset, a reduced version of FB15k with problematic inverse relation pairs removed.

2.5 Related Works

The research paper is heavily inspired by previous works. In the aspect of GCNs for large-scale, highly multi-relational data of realistic knowledge bases, the works of Joan Bruna et al. [6], Michaël Defferrard et. al. [7], and Thomas N. Kipf et al. [3] were considered. In the aspect of R-GCNs as a sub-class of message passing neural networks, the works of Gilmer et al. [2] was considered. The works of Thomas N. Kipf et al. and Gilmer et al. will be briefly discussed with respect to the main research paper.

The main research paper by Schlichtkrull et al. takes heavy inspiration from research paper by Thomas Kipf et al. in terms of entity classification using GCNs. Kipf outlined a similar model for entity classification using $softmax(\cdot)$ activation, and evaluate cross-entropy using:

$$L = -\sum_{l \epsilon Y_L} \sum_{f=1}^{F} Y_{lf} ln\ Z_{lf} \tag{7}$$

Which is identical to the equation used by Schlichtkrull et al. The main difference between the two is in the message-passing framework used in designing the R-GCN.

Gilmer et al. attempts to solve a very different problem than Schlichtkrull et al., predicting the properties of molecules and materials using deep learning. However, we can see that the methodologies used are similar. In fact, the message-passing framework used by Schlichtkrull et al. in the design of the R-GCN was heavily influenced by that of Gilmer et al., which is as follows:

$$m_v^{(t+1)} = \sum_{w \epsilon N(v)} M_t(h_v^t, h_w^t, e_{vw}) \tag{8}$$

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1}) \tag{9}$$

Where the message passing phase runs for T time steps, and hidden states h_v^t at each node are updated based on messages m_v^{t+1}.

2.6 Real World Applications

Node classification and link prediction can be useful in social networks. With the emergence of online social networks, there is an increase in information about individuals, their activities, connections to other individuals or groups and much more. These information are usually modeled as labels associated with individuals. However, such labels are usually inputted by the individuals and may be inaccurate or missing as they may fail to update or add new labels as time progresses. This is where node classification comes into play and helps to update node labels or add missing node labels.

Link prediction can be be useful in social network graphs too. They can be used to give an estimate on the existence of a currently non-existing link. This can be used to give friend recommendations or potential interest group recommendations.

Link and attribute prediction also addresses problems that graph pattern matching seeks to solve; namely query answering over knowledge bases, entity name disambiguation. An example of both query answering and named entity disambiguation can be seen in Figure 1.

Section 3: Comparative Study w/ CZ4071 Techniques & Problems

In this section we offer some of our hypotheses on how the paper's proposed idea might help to solve some of CZ4071 problems. We then suggest some ideas on how to improve the paper's proposed method through innovations from what was taught in CZ4071.

3.1. Solving CZ4071 Problems

Quite intuitively, the main problem addressed in the paper and the course of CZ4071 would be to find a way to solve edge and attribute uncertainty. The author has directly found a method for solving edge and attribute uncertainty in incomplete graphs. In fact, as the R-GCN uses softmax layers, it can be used to generate information on both uncertain and certain graphs - although the representation of uncertain graphs may pose a challenge as the paper uses certain graphs. Similar to classification models with softmax layers, we can use confidence scores for uncertain graphs or compute the max tensor as labels for certain graphs.

In addition to the above solutions the paper brings, we also hypothesize that incomplete graphs can benefit by undergoing edge predictions prior to performing graph pattern matching for critical tasks such as malware detection. While the introduction of edge prediction by the model may create overheads and false positives, it is often acceptable, as an undetected (false negative) malware can be devastating to a system and/or business.

We also believe that R-GCNs can be a helpful tool to solve the influence maximization problem, especially on networks where the node attribute matters. As R-GCNs functions as encoders for a network, the forward pass of the model essentially encodes information of neighbouring and self-node attributes. We can then emphasize key attributes by placing a higher weight, or giving all attributes 0 if the attributes do not matter. Assuming we are trying to computing the influence maximization with an emphasis on attribute values (i.e. a customer with little connections but who is incredibly wealthy that we believe that heavily influence the network), we can use the R-GCN as an encoder to find the top-k tensors (nodes) of high influence nodes.

Finally, we hypothesize that the technique of edge prediction can be used to answer the biggest question throughout the first half of CZ4071 - how do we create graphs that represent real life networks?

To accomplish this, we can first encode a real life network with n number of nodes using the R-GCN. After which, we create a network with the same n nodes (and their attributes) but without edges between them. Applying the R-GCN and DistMult method, we predict the edges of each node in the new network. We believe that on network graphs where the proposed method in the paper excelled in, we would get relatively similar graphs (similar clustering coefficient, degree exponent, degree distribution etc.). This could imply that studies on real life networks might require significantly less data to generate and simulate.

3.2. Techniques to Solve or Improve the Paper's Solution

The R-GCN proposed in the paper uses the sum of adjacent nodes and self-loop attributes as features in the forward pass. We propose including the eigenvector centrality and the PageRank score of each node as additional features for the model. We hypothesize that this might allow the model to better represent the graph structure for encoding.

Of the four centrality measurements taught in CZ4071, we believe that a node's eigenvector centrality is the most relevant centrality measurement. Unlike the other forms of centrality measurements, eigenvector centrality accounts for both the immediate number of edges to a node, as well as the importance of neighbouring node edges throughout the network. Computing the PageRank score of a node can also establish its prominence as a hub or authority.

With the PageRank score and eigenvector centrality score included as features, the model might better understand the behaviour of nodes in the graph, and might give a glimpse into the likelihood of two nodes connecting. The prominence of a node as a hub might indicate that it is likely

to establish connection with a prominent authority node, and vice versa. Finally, the eigenvector centrality and PageRank scores can be derived relatively conveniently through the adjacency matrix which would have already been used in the R-GCN implementation (albeit we might have to swap the variations of neighbouring nodes' attribute weights with 1's to indicate an edge).

Section 4: Implementation

The paper has shown that using the RGCN has produced state-of-the-art results for Node/Entity Classification and Link Prediction. Experiments were conducted to verify these results and to gather insights on the type of data that is suitable for these tasks as well as the difficulty of implementing the solution.

4.1 Entity Classification

Table 1: Detailed Results of Node Classification Experiments

	AIFB	MUTAG	BGS	AM	New Data (CORA)
RGCN (Paper)	95.83	73.23	83.1	89.29	-
RGCN (Implemented)	97.22	72.06	86.21	OOM	79.01

The result screenshots are attached in Appendix B. As mentioned in the paper, MUTAG and BGS performs the worst due to the nature of the dataset. Our implemented RGCN attained a higher score of 97.22% on the AIFB dataset but it might be due to lucky initialisation of weights. The AM dataset produced an Out Of Memory Exception on CPU as well as GPU due to the dataset being too large as it has 1,666,764 entites and 5,988,321 edges. From the embeddings in appendix A , we can see that aside from a slight overlap the classes are well separated. This indicates that the model performs well at clustering the nodes into the correct classes.

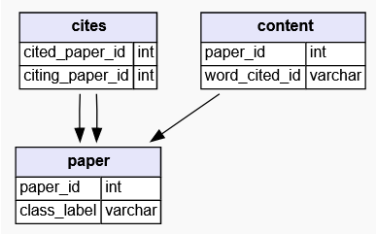

Figure 5: Example of Cora dataset

The Cora dataset consists of 2708 scientific publications classified into one of seven classes and the citation network consists of 5429 links. The performance of RGCN on Cora is slightly lower than BGS. Labeled nodes in BGS are connected via high-degree hub nodes that encode a certain feature. RGCN is known to perform poorly due to the fixed choice of normalization constant for the aggregation of messages from neighboring nodes, which is problematic for nodes of high degree. Given that Cora is a scientific paper citation network, it might have a few high degree nodes since a few papers are cited very frequently, for example the current paper we are researching. Hence, we can observe similar results to BGS. From the rankings in [8], it can be seen that RGCN places around middle in terms of accuracy among the best models for Cora.

Experiments were conducted to see if training in batches would make it possible for our current hardware to use the AM dataset. Unfortunately, the 6GB GPU memory was still exceeded. We would suggest the F1 score should also be measured by researchers for clearer classification insights.

4.2 Link Prediction

Table 2: Detailed Results of Link Prediction Experiments

	FB15k					WN18				
RGCN+ (Paper)	MRR		Hits @			MRR		Hits @		
	Raw	Filtered	1	3	10	Raw	Filtered	1	3	10
	0.262	0.696	0.601	0.760	0.842	0.561	0.819	0.697	0.929	0.964
RGCN+ (GPU)	OOM									
RGCN + (CPU 20 Epochs)	OOM	OOM	OOM	OOM	OOM	-	0.006	0.0002	0.0057	0.0133
	FB15k-237					Toy				
	MRR		Hits@			MRR		Hits@		
	Raw	Filtered	1	3	10	Raw	Filtered	1	3	10
RGCN+ (Paper)	0.156	0.249	0.151	0.264	0.417	-	-	-	-	-
RGCN+ (GPU)	OOM	OOM	OOM	OOM	OOM	-	0.412	0.245	0.504	0.784
RGCN + (CPU 20 Epochs 12000 for Toy)	OOM	OOM	OOM	OOM	OOM	-	0.425	0.257	0.516	0.796

All of the original datasets had an Out Of Memory Error when training with the GPU. The training time on CPU was estimated to take 42 hours, hence we decided to reduce the number of Epochs to 20. Even after this step, the 16GB RAM and Swap were full for the datasets FB15k and FB15k-237 and the training could not be completed. A high Mean Reciprocal Rank (MRR) and Hits@n score, indicates a better performance of the model since MRR and Hits@n in this case measures correlation of ranking of the link prediction compared to the ground truth ranking. Link prediction on the Toy dataset produces good results when compared to the paper results of the FB15k-237 dataset. This is surprising as the FB-Toy dataset is a modified subset of the FB15k-237 dataset and was created to reduce memory consumption while training. The FB-Toy dataset consists of triples like athlete to (country,sport,awards) and only contains 279 entities. These results show promise in predicting relations between entities and it is possible that in the future, information retrieval systems like Google will be able to leverage this even more to retrieve near complete information about an entity.

References

[1] Michael Schlichtkrull et al. *Modeling Relational Data with Graph Convolutional Networks*, 2017. arXiv:1703.06103.

[2] Gilmer et al. *Neural Message Passing for Quantum Chemistry*, 2017. arXiv:1704.01212.

[3] Thomas N. Kipf, Max Welling. *Semi-Supervised Classification with Graph Convolutional Networks*, 2017. arXiv:1609.02907 =

[4] Yang et al. *Embedding Entities and Relations for Learning and Inference in Knowledge Bases*, 2014. arXiv:1412.6575.

[5] Trouillon et al. *Complex Embeddings for Simple Link Prediction*, 2016. arXiv:1606.06357.

[6] Bruna et al. *Spectral Networks and Deep Locally Connected Networks on Graphs*, 2014. arXiv:1312.6203.

[7] Defferrard et al. *Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering*, 2017. arXiv:1606.09375.

[8] CORA. *Relational Dataset Repositry*, https://relational.fit.cvut.cz/dataset/CORA

Appendix A

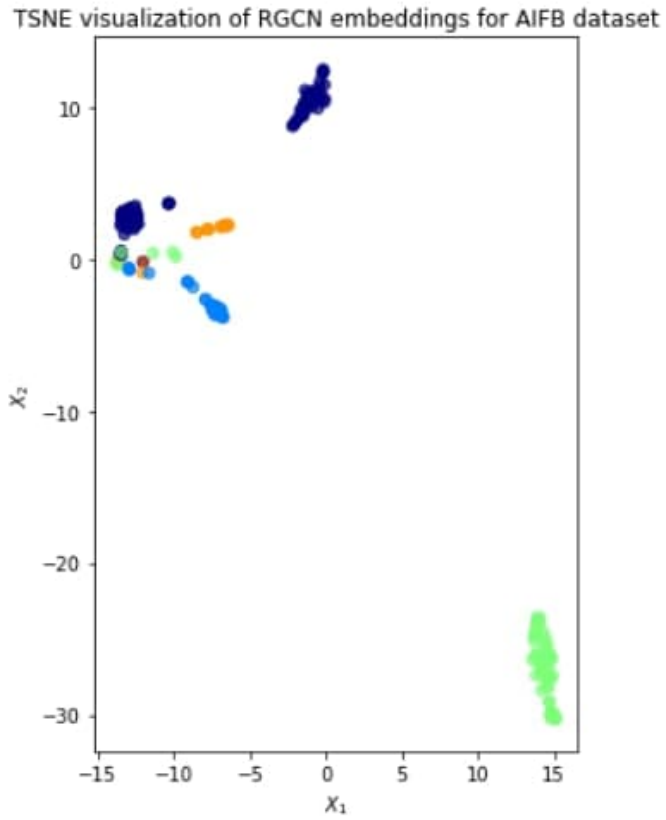

Figure 6: TSNE of RGCN embeddings on AFIB dataset

Appendix B

Datasets	Results
AIFB	Test set results: loss= 0.1086 accuracy= 0.9722
MUTAG	Test set results: loss= 0.5865 accuracy= 0.7206
BGS	Training is complete! Starting evaluation... [Evaluation] Test Accuracy: 86.21 [Summary] Test Accuracy: 86.21 -/+ 0.00 INFO - R-GCN Node Classification - Completed after 0:03:48
CORA	Test Set Metrics: loss: 1.6508 acc: 0.7901
WN18	[Final Scores] Total Epoch 20 MRR(filtered): 0.005685141111271829 Hits@1(filtered): 0.0002 Hits@3: 0.0057 Hits@10(filtered): 0.0133 INFO - R-GCN Relation Prediction - Completed after 0:28:53
Toy	[Final Scores] Total Epoch 12000 MRR(filtered): 0.4252357100512723 6 Hits@1(filtered): 0.25 Hits@3(filtered): 0.5164473684210527 Hits@10(filtered): 0.7960526315789473 INFO - R-GCN Relation Prediction - Completed after 0:22:52

Figure 7: Screenshots of Results