Link Prediction and Node Classification in Cora Citation Network

Swathi Doraiswamy Gopal
Rutgers University- New Brunswick
sd1322@scarletmail.rutgers.edu

Arathi Reghukumar
Rutgers University- NewBrunswick
ar1778@scarletmail.rutgers.edu

ABSTRACT

Network Graphs have evolved in every discipline, and there is a natural increase in predicting tasks on graphs. Graph embeddings have shown to be extremely useful for applications ranging from content suggestion to computer-aided medication creation. Using the CORA graph dataset, we investigate a range of approaches to perform node classification and link prediction

KEYWORDS
GraphSAGE, Node Classification, Link Prediction, Logistic Regression

1 INTRODUCTION

Predictions of graphs require a way to properly understand the information and convert it into a mathematically meaningful structure. Node embeddings are extremely useful in graph prediction problems. These node embeddings reduce high-dimensional information about nodes, such as their neighbors, degrees, and so on, to a one-dimensional vector representation. This greatly simplifies a variety of prediction tasks such as node Classification and Link prediction.
Graph instances can be used to depict complex networks. Edges of the graph can reflect complicated interactions between elements. In the actual world, complex networks include online social networks, knowledge graphs, biological networks like genetic interactions or protein-protein interactions, and citation networks. All of these real-world networks are dynamic in nature, with nodes and edges added and withdrawn. There are numerous advantages to studying the trend with which these graphs change.

To make things more explicit, understanding future associations between items and classifying these entities is critical. This is because the problem of link prediction and node classification in complicated networks has been a persistent challenge since the dawn of technology. Link prediction is the challenge of predicting future connections among graph entities given a graph, which is nothing more than an abstraction of entities and their interactions in a given network. The task of classifying items into classes for better comprehension is known as node classification.
We are using the CORA citation dataset, which has 2708 nodes representing scientific publications sorted into one of seven types and 5429 graph edges showing relationships. We implement a model that predicts the class of a node and the existence of a link given two nodes using GraphSAGE node embeddings.

2 RELATED WORK

The machine learning community has conducted substantial research on feature engineering under a variety of topics. The traditional method for producing features for nodes in networks is based on feature extraction techniques, which often entail some hand-crafted seed features based on network parameters.[1,2]. On the other hand, we intend to automate the process by recasting feature extraction as a representation learning issue, in which case no hand-engineered features are required. Unsupervised feature learning algorithms make use of the spectral features of various graph representations employing matrices, particularly the Laplacian and adjacency matrices. These methods can be considered as dimensionality reduction strategies from the standpoint of linear algebra. Eigen decomposition of a data matrix is computationally expensive unless the solution quality is considerably reduced by approximations.As a result, scaling these technologies to large networks is difficult. Furthermore, these methods optimize for objectives that are strong enough to account for the numerous patterns observed in networks, and they make assumptions about the link between the underlying network structure and the prediction task. The Skip-gram model [2] specifically aims to learn continuous feature representations for words by optimizing a neighborhood preservation likelihood target.

Labeled and unlabeled data are represented as vertices in a graph, with the weights of the vertices between them indicating the strength of correlation between them. We may then formalize the problem using a Gaussian random field

Another method is to utilize manifold regularization to incorporate labeled and unlabeled data into the development of a general-purpose trainer. This method's theoretical foundation is based on Reproducing Kernel Hilbert spaces. Using these, the model could then effectively handle unlabeled data. [3]

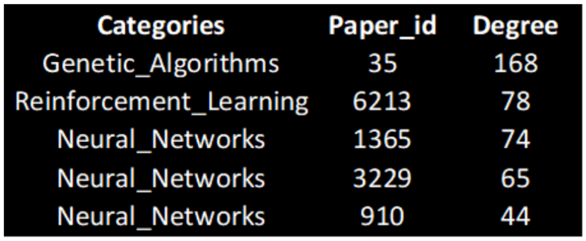
Deep semi-supervised embeddings used for deep multi-layer neural network topologies, either on every layer or just at the output layer, are one significant approach. When compared to the other semi-supervised strategies outlined above, this strategy produces competitive results. [4]

Newer models, on the other hand, tend to focus more on models constructed on top of the skip-gram model [5]. The skip-gram model was unique for not using matrix multiplication to generate its embeddings, allowing it to train up to 100 billion words in a single day. [6]

Finally, algorithms such as DeepWalk [7] and LINE [8] sample random walks on a graph to build embeddings for each node.

3 PRELIMINARIES

We examined the dataset to determine its primary characteristics. The dataset is not a connected network, as we discovered. Figure 1 depicts the top five categories with the highest degrees. The collection has 3563 cliques in total. The largest clique size in this dataset is 5. There are 9 size 5 cliques. They are all based on Reinforcement Learning or Neural Networks. As shown in Fig 2, the class "Neural Networks" has the most scientific publications, followed by "Probabilistic methods," which has the second most scientific papers. Figures 3,4 show the visualization of the CORA citation dataset.



Categories	Paper_id	Degree
Genetic_Algorithms	35	168
Reinforcement_Learning	6213	78
Neural_Networks	1365	74
Neural_Networks	3229	65
Neural_Networks	910	44

Figure 1: Scientific paper with highest degrees

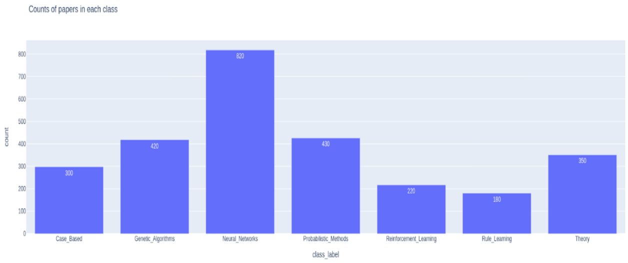


Figure 2: Counts of Paper in Each class

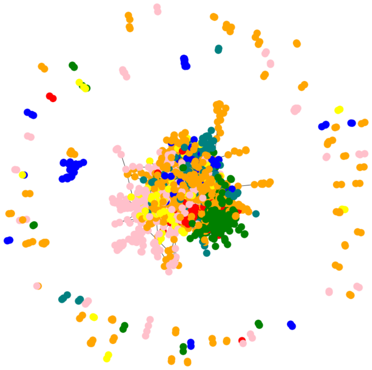


Figure 3:Plot of scientific Paper as nodes and citation links

Figure 4: 3D Plot of Graph

4 PROBLEM FORMALIZATION

4.1 Link Prediction

Every node in the graph is transformed into an embedding. This is a vector with a low dimension. Any node N is represented as a 1xdim vector. Dim is a parameter that we set here.

For the GraphSAGE link prediction model, we use the multiply operator to concatenate the node vectors and obtain a score for the existence of a link between the nodes.

4.2 Node Classification

For this challenge, like with link prediction, each node in the graph is transformed to a low-dimensional vector. The label to which the node belongs, however, remains intact. The embeddings are fed into a single layer of a typical neural network, and the labels (in one-hot vector form) are fed out. The final output layer in a typical multi-class classification problem is a softmax layer, as expected. We use the graphSAGE model to produce an embedding for the node in the context of the graph, which we then input to the predictor of the dense model we designed to categorize previously unknown nodes.

5 THE PROPOSED MODEL

GraphSAGE is a framework for inductive semi-supervised learning. SAGE stands for sampling and aggregation. The architecture generally outperforms GCN by incorporating node representation of not only a node's neighbors, but also the neighbors of its neighbors. It only randomly samples a defined number of neighbors and neighbors of neighbors at each iteration for a fixed number of iterations to save overhead. The model we created is made up of a single layer of a 32x32 GraphSAGE Model whose inputs are the sampled inputs aggregated by the generator. The model then generates vectorized inputs and vectorized outputs that can be sent to a single dense layer of a neural network, the outputs of which are then supplied to a softmax output layer to enable for multi-class classification.

For link prediction, we take the node embedding vector v and perform the following operation to get the link score:

$$score (\hat{y}) = v T . v$$

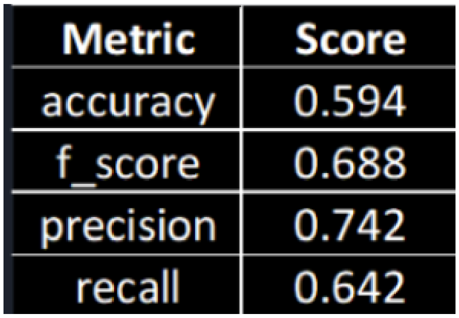
6 EXPERIMENTS

We have evaluated our models against four popular metrics - Accuracy, Precision, Recall and f-score.

6.1 Link Prediction

Overall, we performed link prediction using two alternative models - Logistic Regression and graphSAGE - to produce benchmarks. Because of its simplicity, the Logistic Regression model served as the baseline.

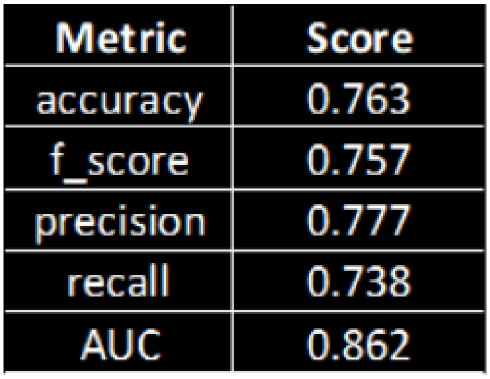
6.1.1 Logistic Regression.
The Node2Vec approach was used to produce embeddings for the simple logistic regression model. The following are the outcomes:



Metric	Score
accuracy	0.594
f_score	0.688
precision	0.742
recall	0.642

6.1.2 GraphSage.
With a kernel size of 16 x 16, a single hidden layer of Graph convolutional network is employed. The result is h, a node representation. We used a dot product between the source and destination node vectors to make the prediction. These are the scores for the link between the source and destination nodes.

We played around with different kernel sizes and hop distances. The metric values for the optimal size 16 × 16 with hop distance 2 are shown below. However, this is not thorough testing, and the outcomes represent a local optimum.



Metric	Score
accuracy	0.763
f_score	0.757
precision	0.777
recall	0.738
AUC	0.862

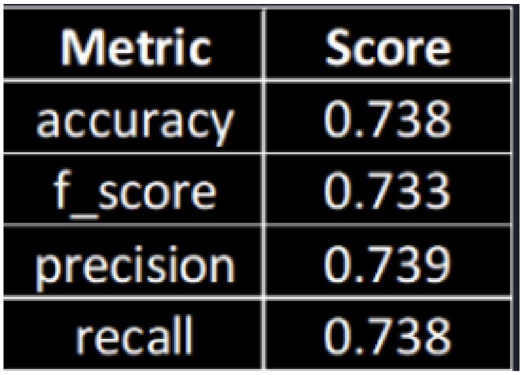
From the combined metric results we noticed that the GraphSAGE algorithm gives better results as compared to Logistic regression.

6.2 Node Classification

Overall, we attempted node classification to produce benchmarks, employing two different models - Logistic Regression and graphSAGE. Because of its simplicity, the Logistic Regression model served as the baseline.

6.2.1 Logistic Regression.

The simple logistic regression model was fed embeddings generated using the Node2Vec algorithm. The results obtained are as follows:

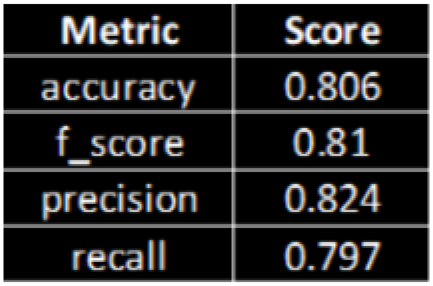


Metric	Score
accuracy	0.738
f_score	0.733
precision	0.739
recall	0.738

6.2.3 GraphSage.

A single hidden layer of Graph convolutional network with kernel size 32 x 32 was employed. A softmax layer was employed to classify the findings.

The number of samples counted was preserved at 10 for the first hop and 5 for the second hop. We tested with numerous settings for this count, hoping for an improvement or a degradation. This was not seen, and the results were all quite comparable to those shown below. We believe this is due to the dataset's short size; altering the count will have a greater influence when dealing with much larger datasets.



Metric	Score
accuracy	0.806
f_score	0.81
precision	0.824
recall	0.797

From the combined metric results we noticed that the GraphSAGE algorithm gives better results as compared to Logistic regression.

7 CONCLUSIONS AND FUTURE WORK

We employed the CORA citation dataset in this experiment, which consists of 2708 nodes representing scientific publications sorted into one of seven types and 5429 graph edges representing relationships.

Several models are implemented that employ GraphSAGE node embeddings to predict the class of a node and the existence of a link given two nodes. We analyzed each model to determine which techniques are best and most appropriate for our link prediction and node classification challenges. We use metrics like accuracy, f score, precision, recall and AUC (Area under curve). Considering these metrics, we found that the GraphSAGE algorithm performs better .

To improve on the current results, we recommend developing a more advanced ensemble model of GraphSAGE, GCN, and other effective models in the future. We seek to improve the ensemble model's accuracy so that we can make more accurate predictions and classifications.
To achieve better node or link embeddings, we intend to improve feature engineering by developing a robust feature transformation and selection approach. To improve the accuracy of the present models, we propose more thorough hyperparameter adjustment. In addition, we intend to improve the data by balancing the samples in

each class. We expect that by making these changes, we will be able to attain better results for our problem.

8 ACKNOWLEDGEMENT

Professor Yongfeng Zhang from Rutgers University provided us with resources and knowledge. We are grateful for all of the assistance and learning opportunities provided by the faculty. This project could not have been achieved without the collaboration of the entire team and the assistance of all the resources cited.

REFERENCES

[1] B. Gallagher and T. Eliassi-Rad. utilizing label-independent features for classification in sparsely labeled networks: An empirical study. Advances in Social Net- work Mining and Analysis. Springer, 2009.
[2] K. Henderson, L. Li, L. Akoglu, B. Gallagher, T. Eliassi Rad, H. Tong, and C. Faloutsos. Itaˆs who you know: graph mining
[3] Zhu et al. 2003. Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. ICMl (2003)
[4] Belkin et. al. 2006. Manifold Regularization: A Geometric Framework for Learn- ing from Labeled and Unlabeled Examples. JMLR (2006).
[5] Weston et al. 2012. Deep Learning via Semi-Supervised Embedding. IGAR (2012).
[6]Mikolov et. al. 2013. Distributed Representations of Words and Phrases and their Compositionality. arvix (2013).
[7] Perozzi et al. 2014. DeepWalk: Online Learning of Social Representations. arxiv (2014).
[8] Tang et al. 2015. LINE: Large-scale Information Network Embedding. arxiv (2015).