Modeling Citation Networks for Paper Classification

Hwai-Jin Peng
Department of Electrical & Computer Engineering
University of Washington
hjpeng@uw.edu

Cheng-Yen Yang
Department of Electrical & Computer Engineering
University of Washington
cycyang@uw.edu

1 Introduction

Citations have several important purposes: to uphold intellectual novelty, to avoid plagiarism, and to justify or extend the knowledge from prior works. Therefore, it is of great importance to understand how knowledge is passed on by studying the patterns that emerge in the citation flow. A general approach is to categorize earlier works of the same problem for review. However, such a large scale of publications makes finding relevant researches from the number of articles a nontrivial problem. Thus, an automated classifier for research citations (CRC) is needed.

A citation network is composed of a collection of publications that are linked by directed edges representing the citation relationships from one publication to another. Most of the existing works on the citation classification fall into three categories: **1) content-based, 2) relation-based,** and **3) graph-based methods**. Content-based approaches utilize the features of the document itself, such as keywords, embedded representations of title or abstract, to predict the research field of the paper, while the relation-based methods perform classification based on the information of the neighboring nodes (i.e. publications). A graph-based model takes the entire citation network as a directed, acyclic graph and leverages both node features and edge information to classify the paper topic.

In this work, we do a quick review of some popular content- and relation-based classifiers in Section 2 and introduce the CoRA dataset in Section 3. Then we present our graph-based model, a supervised graph convolutional network (GCN), to classify papers regarding both paper content and relationships along with other baseline models in Section 4. We show our experiment results in Section 5. And finally, we discuss and summarize our findings in Section 6.

2 Relative Works

It is important to obtain proper features from publications as they are meaningful representations of documents. Skip-gram [1] and TF-IDF algorithm [2] were introduced to embed natural language content into low-dimension vectors. After extracting the features of the papers, we can use Naive Bayes classifier or neural networks to get promising results of node (e.g. publication) classification or link (e.g. co-authorship) prediction tasks.

A citationship between the citing and cited publications suggests that they share a similar topic. It is natural to use the structure of the citation network to predict the paper's field. Methods such as majority voting [3] and wvRN [4] were carried out to determine the class of the target node by exploring and observing its neighbors.

Models that jointly reason about the network structures and the content of documents have been intensively applied to classification problems. An effective approach for incorporating the citation relationship into paper content features is representation learning. Random walk based methods were combined with a popular language embedding model to generate better representations of vertices. DeepWalk [5], for instance, treats truncated random walks from the graph as sentences and then leverages an n-gram language model to give high-quality node features for classifiers. VERSE [6], on the other hand, uses a learned embedding matrix and vertex similarity encoder in a PageRank-like random walk. Besides, the node2vec [7] framework learns low-dimensional representations for nodes by optimizing a neighborhood preserving objective.

University of Washington: CSE 547 Machine Learning for Big Data (Spring 2020).

Table 1: Summary of the two datasets.

	CoRA-Raw	CoRA-Small
Nodes	11881	2708
Edges	33416	5278
Isolated Nodes	410	0
Average Degree	5.6251	3.8981

	CoRA-Raw	CoRA-Small
CC	833	78
Features	9568	1432
Classes	10	7

Recently, deep learning models on graphs have received increasing attention. Graph neural networks such as GCNs effectively utilize the node features and network topology to construct meaningful node representations and therefore boosts the classification performance.

3 Dataset

3.1 Dataset Overview

CoRA, a collection of academic papers in the field of computer science, is composed of a set of entities and their relations. Entities are authors and scientific papers. It assigns each paper to a set of categories (multi-label classification task), selected from a taxonomy of classes. CoRA contains 11881 papers belonging to 80 different categories, 16114 authors, and 34648 citations relations between papers. We followed two different approaches [8, 9] to construct two datasets: CoRA-Raw and CoRA-Small. For both two datasets, only the representation vector, class, and citations of each paper were included. The information of authorships was not taken into consideration in either [8] or [9].

In CoRA-Raw, we reduced the number of classes by merging sub-classes into 10 main classes: *AI*, *DAT*, *Networking*, *HA*, *Programming*, *HCI*, *IR*, *OS*, *Databases*, and *EC*. Each paper was associated with a feature vector $v \in \mathbb{R}^{9568}$ containing its title represented as bag-of-words with TF-IDF weights.

In CoRA-Small, we only picked citations belongs to the classes under ML (one of the sub-classes of AI), i.e., *CB*, *GA*, *NN*, *PM*, *REL*, *RUL*, and *Theory*. In addition, the dimension v of the feature vectors of title was reduced to 1432 and became binary to indicate whether this term was in the paper title or not.

3.2 Dataset Summary

The summary and the class distribution of CoRA-Raw and CoRA-Small are shown in Table. 1 and Figure. 1, respectively, where CC denotes the connected components. Note that the class names of Programming, Networking, and Databases in CoRA-Raw were abbreviated as PROG, NET, and DB, respectively. From the class distribution, we can observe that the number of data is imbalanced between classes. Therefore, we will discuss the effect of class imbalance on classification accuracy later.

For further network analysis, we visualize our two networks in Figure. 4 where different color stands for a different class. One can tell that most of the papers cite other papers under the same class. However, from the overlapped clusters in the center in both graphs, we observe that papers in different classes are often closely related. Intuitively, researchers usually extend their frameworks from existing methods that are robust and fundamental. For example, depth-first search algorithms are well-known in *Programming*, but it's widely used in the field of *Networking* as well. Such paper acts like a root in the citation network and thus is cited by papers from different classes. As a result, rather than classifying citations using solely content features or relationships, we can expect that the performance would be improved if we leverage both information.

4 Approaches

In this section, we first present the content-based and relation-based approaches that were commonly used in citation classification problems. Then we show the implementation details of our graph-based model.

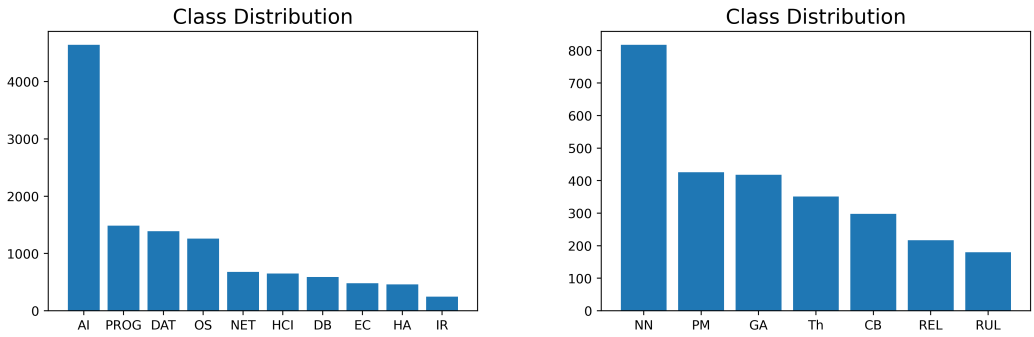

Figure 1: Class distribution of CoRA-Raw (Left) and CoRA-Small (Right).

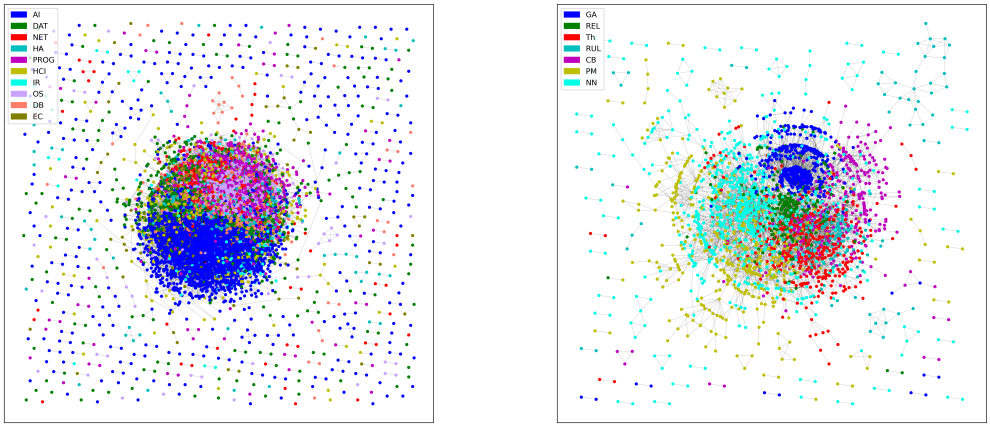

Figure 2: Visualization of CoRA-Raw (Left) and CoRA-Small (Right).

4.1 Content-Based Classification

In general, a content-based classifier considers the profile or the metadata that is associated with the publication. The title, keywords, author name, year of publication, publication venue, etc., are widely used as features to identify the class of the paper.

4.1.1 Naive Bayes Classifier (NBC)

Naive Bayes classifier is a statistical classification technique based on Bayes' Theorem. It's "naive" because it makes an assumption called class-conditional independence that features are mutually independent even if they are related. Given an instance, represented by a feature vector $\mathbf{F} = (f_1, ..., f_n)$ which contains n independent features, the Naive Bayes model assigns this instance to one of the k classes, C_k, with a probability $\mathbb{P}(C_k|\mathbf{F})$. Applying Bayes' theorem, the conditional probability can be decomposed as

$$\mathbb{P}(C_k|\mathbf{F}) = \frac{\mathbb{P}(C_k)\mathbb{P}(\mathbf{F}|C_k)}{\mathbb{P}(\mathbf{F})}. \tag{1}$$

This posterior probability, using the conditional independence assumption and chain rule, can be written as

$$\mathbb{P}(C_k|\mathbf{F}) = \mathbb{P}(C_k|f_1, ..., f_n) \propto \mathbb{P}(C_k) \prod_{i=1}^{n} \mathbb{P}(f_i|C_k). \tag{2}$$

Here, we applied the maximum a posterior (MAP) decision rule, which is commonly used in Bayes model, to assign the instance to the class $\hat{y}$ with the highest probability as follows:

$$\hat{y} = \underset{k}{\operatorname{argmax}}\, \mathbb{P}(C_k|\mathbf{F}), \tag{3}$$

Table 2: Architecture of our neural network. (Left: CoRA-Raw. Right: CoRA-Small)

Layer	Output Size
Input	9568
Fully Connected	128
ReLU + Dropout	128
Output	10

Layer	Output Size
Input	1432
Fully Connected	100
ReLU + Dropout	100
Output	7

where $\mathbf{F}$ denotes the feature vectors of the paper title in our dataset. For CoRA-Small, we used the multivariate Bernoulli kernel since the feature vectors were binary, while for CoRA-Raw, a multinomial kernel was applied due to the data contains continuous TF-IDF values.

4.1.2 Neural Network (NN)

An artificial neural network consists of millions of neurons that are densely connected and usually organized into layers of nodes. A node combines input data with a set of coefficients. A neuron is fired when it encounters sufficient stimuli, thereby determining if the signal should be propagated further through the network to affect the final output, e.g., predictions of class in the classification problem.

Our neural network contained one hidden layer, followed by a ReLU activation layer and then a dropout layer with a 50% dropout rate. The implementation details are shown in Table. 2. The model was trained by an SGD optimizer with an initial learning rate of 0.1 for 80 epochs. For CoRA-Small, the scheduler decayed the learning rate by a factor of 10 for every 15 epochs. While for CoRA-Raw, the learning rate was reduced by a factor of 10 when there was no improvement in the training accuracy for a consecutive 5 epochs. Again, the model took as input the paper title features.

4.2 Relation-Based Classification

Algorithm 1: Random Walk

Input: Graph $\mathcal{G}$, Test Node v_t, Path Length h, Repeat Times t
Output: The predicted class $\hat{y}$ of node v_t

$\mathcal{C} := \emptyset$; // occurrence counter of class label
for i = 1 *to* t **do**
  Randomly walk for h steps, starting from v_t
  $\mathcal{V}_n :=$ nodes on the path of random walk
  for $v_n \in \mathcal{V}_n$ *and* $v_n \notin \mathcal{V}_t$ **do**
    $c_n :=$ the class that v_n belongs to
    $\mathcal{C}[c_n] \mathrel{+}= 1$; // update the occurrence counter
  end
end
$\hat{y} = \text{argmax}_n \mathcal{C}[c_n]$; // Choose the most frequent class
return *The predicted classes* $\hat{y}$

Algorithm 2: Majority Voting

Input: Graph $\mathcal{G}$, Test Node v_t, Hop Distance k
Output: The predicted class $\hat{y}$ of node v_t

$\mathcal{C} := \emptyset$; // occurrence counter of class
$\mathcal{V}_n :=$ nodes within the k-hop neighbors of v_t
for *Each* $v_n \in \mathcal{V}_n$ **do**
  $c_n :=$ the class that v_n belongs to
  $\mathcal{C}[c_n] \mathrel{+}= 1$; // update the occurrence counter
end
$\hat{y} = \text{argmax}_n \mathcal{C}[c_n]$; // Choose the most frequent class
return *The predicted classes* $\hat{y}$

Relation-based approaches predict the field of study of the publication by traversing through the citation network to explore the relationships between the citing and cited publications. A paper is cited by another that often suggests they are more likely to have similar content or background.

4.2.1 Random Walk (RW)

Similar to MV, the random walk approach assigns the node v to the dominant class within its neighborhood. RW, however, records the occurrence of each class that appears in a path generated by a random walk process. Starting from node v, we traversed through the citation network for h steps to explore its neighbors. This step was repeated for t times, and thereby a total of $h \times t$ nodes were encountered during the process. After that, the node v would be assigned to the most frequent class in the set of visited nodes. If all of the class labels of nodes on the path were unknown, i.e., they were all nodes in the testing set, we assigned v to the dominant class in the entire network. The algorithm is shown in Algorithm 1.

4.2.2 Majority Voting (MV)

The majority voting algorithm is a relation voting method [4]. Given a node v (publication), a counter records the number of occurrences of each class that appears in the k-hop neighborhood of node v. Then, node v will be assigned to the class with the highest occurrence frequency. If there was a tie-break, we chose the label which had a higher occurrence in the training dataset. The algorithm is shown in Algorithm 2.

4.3 Graph-Based Classification

In graph-based models, the entire citation network is viewed as a graph where each vertex stands for a document and each edge represents a citation from the current publication to another. When predicting, the model considers the content of the document as well as the relationship between the neighboring publications and the document itself.

4.3.1 Graph Convolutional Network (GCN)

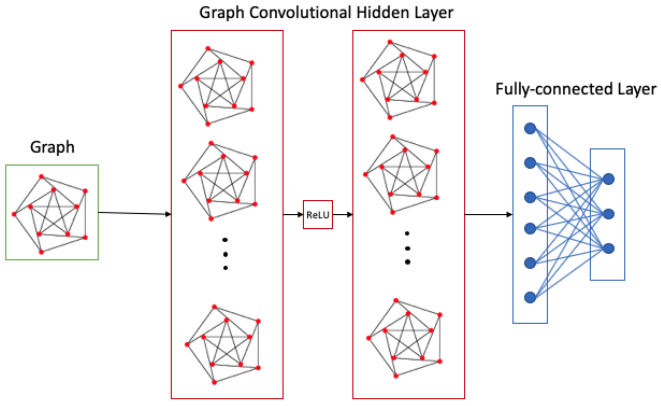

Figure 3: Architecture of our two-layer Graph Convolutional Network (GCN).

As in [10], Kipf et. al. proposed a scalable approach for semi-supervised learning on graph data using convolutional neural networks which operate directly on graphs called Graph Convolutional Networks(GCNs).

The ultimate goal of GCN is to learn a mapping that the input contains a feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and a representative matrix of the graph $\mathbf{A} \in \mathbb{R}^{n \times n}$ (n as the number of nodes in the graph and d is the dimension of features of the nodes) to generate a node-level output $\mathbf{Z} \in \mathcal{R}^{n \times f}$ (f as the output dimension depending on the task).

Here, we make a quick review on how GCN works and its training process using the example from [10]. Given a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ and its corresponding feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and symmetric adjacency matrix

$\mathbf{A} \in \mathbb{R}^{n \times n}$. Our model is a two-layer architecture with the first-layer weights $W_{in} \in \mathbb{R}^{C \times H}$ and the second-layer weights $W_{out} \in \mathbb{R}^{H \times F}$. The feed-forward function becomes:

$$\mathbf{Z} = softmax\Big(\hat{\mathbf{A}} ReLU\big(\hat{\mathbf{A}}\mathbf{X}W_{in}\big)W_{out}\Big), \tag{4}$$

where $ReLU$ and $softmax$ is the activation of W_{in} (input-to-hidden weights) and W_{out} (hidden-to-output weights), respectively. Since our task is a semi-supervised classification, we evaluated and back-propagated using the cross-entropy loss over training samples:

$$\mathcal{L} = -\sum_{i \in \mathcal{Y}_t} \sum_{j=1}^{F} \mathbf{Y}_{ij} ln\big(\mathbf{Z}_{ij}\big), \tag{5}$$

where $\mathcal{Y}_t$ represents the training data while $\mathbf{Y}$ and $\mathbf{Z}$ stands for the ground truth labels and predicted labels, respectively. The GCN weights W_{in} and W_{out} were trained and updated using a stochastic gradient descent optimizer.

We built and trained our GCN on CoRA-Raw and CoRA-Small following the architecture in [10] using the SplineCNN layer introduced by [11]. The hidden dimension for each model was set to 128 and 32. Both of the models were trained with an Adam optimizer with an initial learning rate of 0.01 and a weight decay of 0.005 then recorded the final accuracy after 200 and 500 iterations for CoRA-Small and CoRA-Raw, respectively.

5 Results

In section 5.1, we demonstrate the multi-class classification accuracy of each model we presented in the last section. For both CoRA-Raw and CoRA-Small, we chose four different ratios, i.e., 80/20, 60/40, 40/60, and 20/80, to split the dataset into training and testing set. Then, in section 5.2, we make discussions on the factors that affect the performance.

5.1 Quantitative Results

5.1.1 Content-based Classifiers

From Table. 3 and 4, it's clear to tell that the accuracy increased when the proportion of the labeled data became larger. The model will have a better generalization ability if it is trained with a bigger training set. However, the way to choose features to represent a document has a great impact on the performance, especially for large scale datasets like CoRA-Raw.

Recalls that the Naive Bayes classifier assumes that all features are independent. But, in the CoRA dataset, it's not the case. The features contain TF-IDF values of the title of publications, and therefore some words are likely to appear together. For instance, the term "neural" is often followed by "network" in the paper under the *ML* class. As a result, this assumption hurts the performance and the algorithm becomes less accurate when the dimension of features grows since it will be less likely to have independent predictors.

On the other hand, the neural networks have difficulties when training with sparse inputs like the TF-IDF features. While the neural networks are good at generalizing, they have to accumulate across many variations within the dataset before being robust. If the data is highly sparse, the network learns nothing since most of the data along each dimension are zero. And that's exactly the situation in our CoRA-Raw.

Table 3: Classification results of the Naive Bayes classifier.

Labeled Nodes (%)	20%	40%	60%	80%
CoRA-Raw	41.26%	44.59%	48.18%	**51.07%**
CoRA-Small	53.76%	71.08%	74.82%	**76.20%**

Table 4: Classification results of the neural network.

Labeled Nodes (%)	20%	40%	60%	80%
CoRA-Raw	58.46%	62.46%	64.76%	**66.64%**
CoRA-Small	67.01%	71.08%	74.08%	**76.83%**

5.1.2 Relation-based Classifiers

From Table. 5 and 6, we can also see the trend that the accuracy increased as the percentage of the given labeled samples rose during training. Besides that, we also conducted experiments on different path length h and hop distance k for our random walk and majority voting, respectively.

In our random walk, we can observe a very different behavior using different ratios of the train/test split on both CoRA-Raw and CoRA-Small. As the path length h increased, the result converged to some certain values as the length had surpassed or got closer to the diameter graph. In addition, the performance was unstable since we might encounter nodes that were cited by the target node indirectly, but under different classes and had a great difference in content.

Table 5: Classification results of the random walk with distinct path length h and repeated time $t = 3$.

Labeled Nodes (%)		20%	40%	60%	80%
CoRA-Raw	RW ($h = 1$)	33.87%	51.25 %	61.13 %	**67.21%**
	RW ($h = 3$)	41.20%	55.82%	**61.96%**	65.73%
	RW ($h = 5$)	**44.81%**	**56.78%**	61.15%	63.90%
CoRA-Small	RW ($h = 1$)	40.07%	57.76%	68.33%	**76.34%**
	RW ($h = 3$)	47.80%	63.85%	**69.81%**	75.29%
	RW ($h = 5$)	**51.12%**	**64.05%**	69.18%	73.09%

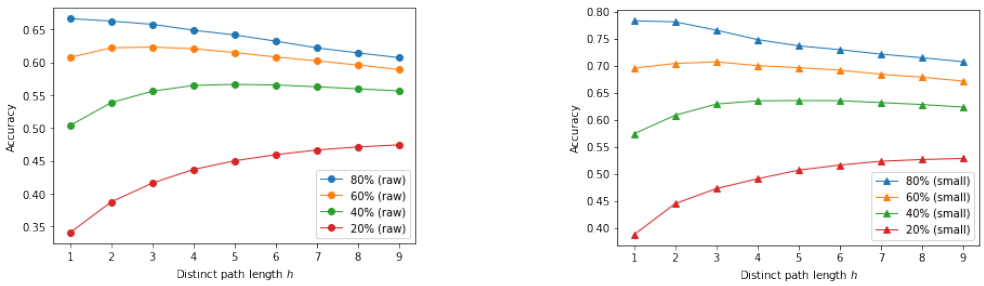

Figure 4: Plots of random walk method using distinct h on different train/test split on CoRA-Raw (Left) and CoRA-Small (Right).

For majority voting, we observe that when the distance hop k was set to 1, the model achieved a higher accuracy if more labeled nodes were given. In contrast, a larger distance hop achieved better accuracy when less labeled nodes were provided as it's more likely to encounter unlabeled nodes in the vicinity of the target node. Majority voting is surprisingly effective and straight-forward especially on the performance for CoRA-Raw as it achieved a very high accuracy of 76.48% on an 80% train/test split.

Table 6: Classification results of the majority voting with distinct hop distance k.

Labeled Nodes (%)		20%	40%	60%	80%
CoRA-Raw	MV ($k=1$)	60.31%	69.81%	74.64%	**76.48%**
	MV ($k=2$)	**70.20%**	**73.65%**	**74.89%**	74.55%
	MV ($k=3$)	66.36%	66.95%	66.94%	67.73%
CoRA-Small	MV ($k=1$)	49.53%	64.96%	76.26%	**83.54%**
	MV ($k=2$)	71.14%	**78.02%**	**81.80%**	83.17%
	MV ($k=3$)	**71.79%**	75.18%	77.65%	81.51%

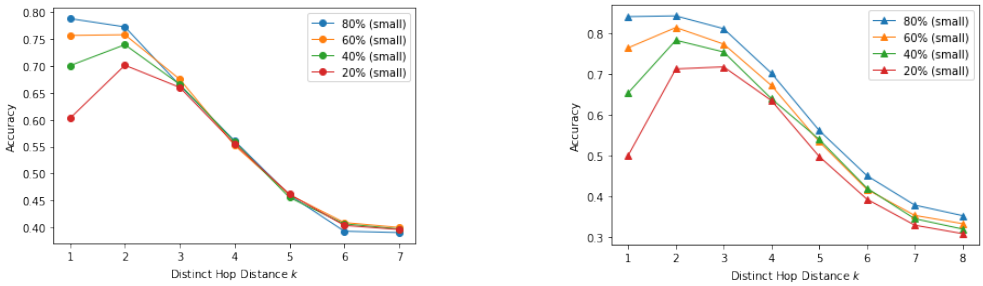

Figure 5: Plots of majority voting method using distinct k on different train/test split on CoRA-Raw (Left) and CoRA-Small (Right).

5.1.3 Graph-based Classifiers

From Table. 7, we observe that with different percentages of labeled nodes, the GCN was less likely to be affected by the number of labeled nodes provided. The accuracy dropped slightly when reducing the ratio of training samples. It indicates that the GCN structure was able to learn and classify with only a small proportion of information from the structured graph.

We also found that the dimension of hidden layers played an important role in the fine-tuning process as it affected the final results. Due to the number of input features and nodes are distinct between two dataset, it made our GCN for CoRA-Raw has nearly 20 times more parameters in comparison to the one for CoRA-Smalland led our model to lose the great generalization ability on CoRA-Raw. We will have a more detailed discussion in the following section regarding GCN on CoRA-Raw.

Table 7: Classification results of the graph convolutional network.

Labeled Nodes (%)	20%	40%	60%	80%
CoRA-Raw	62.34%	64.40%	67.43%	**74.14%**
CoRA-Small	84.67%	87.13%	87.75%	**88.35%**

5.2 Discussions

5.2.1 Class Imbalance

From Figure. 1, *AI* and *NN* is the dominant class in CoRA-Raw and CoRA-Small, respectively. This is bad for our classifier as the model may learn to just predict everything to be that dominant class. Thus, there's a higher chance for our models to wrongly predict an instance to the class *AI* (the first column of confusion matrix in Figure. 6) or *NN* (the last column of confusion matrix in Figure. 7).

5.2.2 Content-based

Most of the time, it is easy to determine the research field of a publication according to the topic. Yet, sometimes it may be too arbitrary to judge one paper by its title. For example, the term "depth-first search"

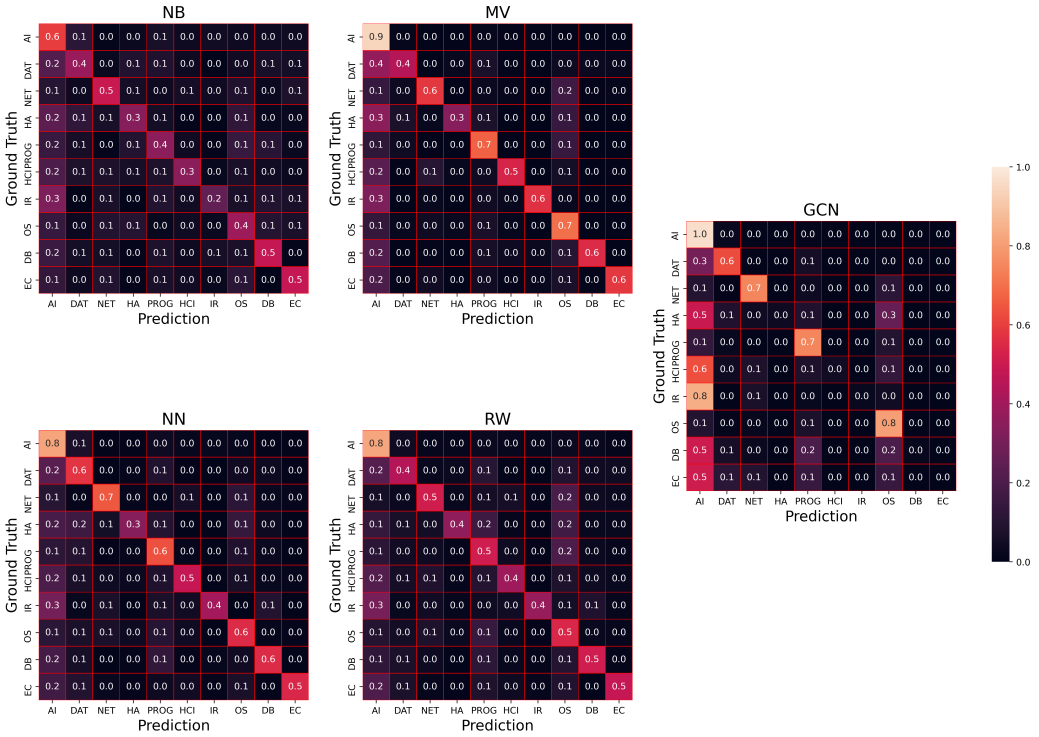

Figure 6: Confusion matrices of different approaches on CoRA-Raw. Each column shows methods leverage different kinds of input data. (Left: content-based. Mid: relation-based. Right: graph-based.)

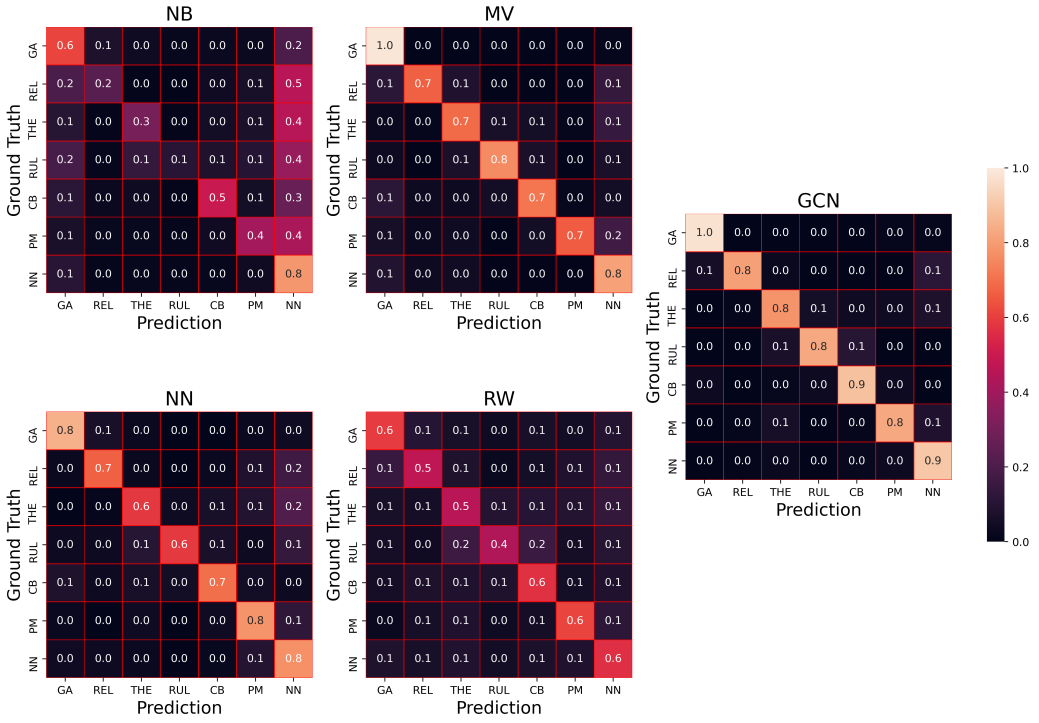

Figure 7: Confusion matrices of different approaches on CoRA-Small. Each column shows methods leverage different kinds of input data. (Left: content-based. Mid: relation-based. Right: graph-based.)

Table 8: Comparisons between all the baseline models and our GCN model.

Labeled Nodes (%)		20%	40%	60%	80%
CoRA-Raw	NB	41.26%	44.59%	48.18%	51.07%
	NN	58.46%	62.46%	64.76%	66.64%
	RW ($h = 3$)	41.20%	55.82%	61.96%	65.73%
	MV ($k = 3$)	**66.36%**	**66.95%**	66.94%	67.73%
	GCN	62.34%	64.40%	**67.43%**	**74.14%**
CoRA-Small	NB	53.76%	71.08%	74.82%	76.20%
	NN	67.01%	71.08%	74.08%	76.83%
	RW ($h = 3$)	47.80%	63.85%	69.81%	75.29%
	MV ($k = 3$)	71.79%	75.18%	77.65%	81.51%
	GCN	**84.67%**	**87.13%**	**87.75%**	**88.35%**

may belong to papers that talk about *Programming*. But it's also widely used in the field of *Networking* or *Robotics*. Thus, we can see some incorrect predictions. In Figure. 6, the confusion matrix suggests that some of the papers of *IR* (Information Retrieval) were assigned to *AI* (Artificial Intelligence), and in Figure. 7, the confusion matrix of NB shows that some of the papers of *RUL* (Rule Learning) or *REL* (Reinforcement Learning) were assigned to *GA* (Genetic Algorithm) when applying content-based approaches.

5.2.3 Relation-based

The confusion matrices of MV and RW in Figure. 6 and 7 were computed using the majority voting algorithm with the hop distance $k = 3$ and the random walk procedure with path length $h = 3$. Relation-based methods assume that publications would cite other papers under the identical class. It's intuitive since lots of works aim to justify or extend from prior studies. Nevertheless, we often find that innovative papers apply knowledge or algorithms from papers belong to other research fields, resulting in a strong connection between different classes. Thus, relation-based approaches are less accurate when classifying works that are close to the boundary of clusters.

5.2.4 Graph-based

On CoRA-Raw, our GCN failed as it overfitted quickly and makes very bias predictions toward the class *AI*. In Figure. 6, we can see that our graph-based model have better results on the labels *DAT*, *NET*, *PROG* and *OS* with accuracy above 0.6. However, we found that our GCN often classified papers that belong to other classes to the dominant class *AI*. It arose from the issue of class imbalance in the dataset. Because we were summing the loss to all the correct classes, the negative log-likelihood term in our cross-entropy loss made our model neglect those low occurrence labels. Thus, it is also important to choose a better objective when dealing with an extremely imbalanced dataset.

6 Conclusions & Future Work

In this work, we presented several popular citation network classifiers using either paper content (i.e., title features) or the citation relationships. Also, we built a graph convolutional network that jointly leveraged the information of network structure and the content of the documents. From the experiments, we found that the graph-based model outperformed other networks that considered only content or relations in terms of classification accuracy. Paper content provides strong connections to a specific class but it would be too arbitrary to rely solely on it, while the citation relationships give us an overview of fields that the paper involves in yet sometimes it is too general. Therefore, the GCNs consider both factors to predict the field of study of the publication from both points of view and achieved a plausible result.

Due to the limitation of the CoRA dataset, we only used the title representations and citationships as our model inputs. Other metadata like the published avenue, year of publication, abstract, and authorships that provide rich information should be included in the future survey. Besides, more research on the relationships between different classes (research fields) is needed to have a deeper understanding of the structure of the citation network.

7 Individual Contributions

- Hwai-Jin Peng
  - Data collection & preprocessing
  - Content-based methods implementation
  - Result analysis
  - Paper writing
- Cheng-Yen Yang
  - GCN implementation
  - Relation-based methods implementation
  - Performance analysis
  - Paper writing

8 Acknowledgment

We would like to thank our TA, Galen Weld, for his valuable advice and suggestion for this final project.

References

[1] Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)

[2] Ramos, J.: Using tf-idf to determine word relevance in document queries (01 2003)

[3] Boland, P.: Majority systems and the condorcet jury theorem. The Statistician **38**, 181 (01 1989). https://doi.org/10.2307/2348873

[4] Macskassy, S.A., Provost, F.: A simple relational classifier. In: Proceedings of the Second Workshop on Multi-Relational Data Mining (MRDM-2003) at KDD-2003. pp. 64–76 (2003)

[5] Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. CoRR **abs/1403.6652** (2014), `http://arxiv.org/abs/1403.6652`

[6] Tsitsulin, A., Mottin, D., Karras, P., Müller, E.: Verse: Versatile graph embeddings from similarity measures. In: Proceedings of the 2018 World Wide Web Conference. pp. 539–548 (2018)

[7] Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864 (2016)

[8] McCallum, A.: Cora Research Paper Classification, `https://sites.google.com/site/semanticbasedregularization/home/software/experiments_on_cora`

[9] Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data. Tech. rep. (2008)

[10] Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)

[11] Fey, M., Lenssen, J.E., Weichert, F., Müller, H.: Splinecnn: Fast geometric deep learning with continuous b-spline kernels. CoRR **abs/1711.08920** (2017), `http://arxiv.org/abs/1711.08920`