Confidence-based Graph Convolutional Networks for Semi-Supervised Learning

Shikhar Vashishth[⋆] **Prateek Yadav**[⋆] **Manik Bhandari**[⋆] **Partha Talukdar**
Indian Institute of Science
{shikhar,prateekyadav,manikb,ppt}@iisc.ac.in

Abstract

Predicting properties of nodes in a graph is an important problem with applications in a variety of domains. Graph-based Semi Supervised Learning (SSL) methods aim to address this problem by labeling a small subset of the nodes as seeds, and then utilizing the graph structure to predict label scores for the rest of the nodes in the graph. Recently, Graph Convolutional Networks (GCNs) have achieved impressive performance on the graph-based SSL task. In addition to label scores, it is also desirable to have confidence scores associated with them. Unfortunately, confidence estimation in the context of GCN has not been previously explored. We fill this important gap in this paper and propose ConfGCN, which estimates labels scores along with their confidences jointly in GCN-based setting. ConfGCN uses these estimated confidences to determine the influence of one node on another during neighborhood aggregation, thereby acquiring *anisotropic*[1] capabilities. Through extensive analysis and experiments on standard benchmarks, we find that ConfGCN is able to outperform state-of-the-art baselines. We have made ConfGCN's source code available to encourage reproducible research.

1 Introduction

Graphs are all around us, ranging from citation and social networks to knowledge graphs. Predicting properties of nodes in such graphs is often desirable. For example, given a citation network, we may want to predict the research area of an author. Making such predictions, especially in the semi-supervised setting, has been the focus of graph-based semi-supervised learning (SSL) (Subramanya and Talukdar, 2014). In graph-based SSL, a small set of nodes are initially labeled. Starting with such supervision and while utilizing the rest of the graph structure, the initially unlabeled nodes are labeled. Conventionally, the graph structure has been incorporated as an explicit regularizer which enforces a smoothness constraint on the labels estimated on nodes (Zhu et al., 2003; Belkin et al., 2006; Weston et al., 2008). Recently proposed Graph Convolutional Networks (GCN) (Defferrard et al., 2016; Kipf and Welling, 2016) provide a framework to apply deep neural networks to graph-structured data. GCNs have been employed successfully for improving performance on tasks such as semantic role labeling (Marcheggiani and Titov, 2017), machine translation (Bastings et al., 2017), relation extraction (Vashishth et al., 2018b; Zhang et al., 2018), document dating (Vashishth et al., 2018a), shape segmentation (Yi et al., 2016), and action recognition (Huang et al., 2017). GCN formulations for graph-based SSL have also attained state-of-the-art performance (Kipf and Welling, 2016; Liao et al., 2018; Veličković et al., 2018). In this paper, we also focus on the task of graph-based SSL using GCNs.

GCN iteratively estimates embedding of nodes in the graph by aggregating embeddings of neighborhood nodes, while backpropagating errors from a target loss function. Finally, the learned node embeddings are used to estimate label scores on the nodes. In addition to the label scores, it is desirable to also have confidence estimates associated with them. Such confidence scores may be used to determine how much to trust

[⋆] Equal contribution

[1] anisotropic (adjective): varying in magnitude according to the direction of measurement (Oxford English Dictionary)

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

the label scores estimated on a given node. While methods to estimate label score confidence in non-deep graph-based SSL has been previously proposed (Orbach and Crammer, 2012), confidence-based GCN is still unexplored.

In order to fill this important gap, we propose ConfGCN, a GCN framework for graph-based SSL. ConfGCN jointly estimates label scores on nodes, along with confidences over them. One of the added benefits of confidence over node's label scores is that they may be used to subdue irrelevant nodes in a node's neighborhood, thereby controlling the number of effective neighbors for each node. In other words, this enables *anisotropic* behavior in GCNs. Let us explain this through the example shown in Figure 1. In this figure, while node a has true label L_0 (white), it is incorrectly classified as L_1 (black) by Kipf-GCN (Kipf and Welling, 2016)[2]. This is because Kipf-GCN suffers from limitations of its neighborhood aggregation scheme (Xu et al., 2018). For example, Kipf-GCN has no constraints on the number of nodes that can influence the representation of a given target node. In a k-layer Kipf-GCN model, each node is influenced by all the nodes in its k-hop neighborhood. However, in real world graphs, nodes are often present in *heterogeneous* neighborhoods, i.e., a node is often surrounded by nodes of other labels. For example, in Figure 1, node a is surrounded by three nodes (d, e, and f) which are predominantly labeled L_1, while two nodes (b and c) are labeled L_0. Please note that all of these are estimated label scores during GCN learning. In this case, it is desirable that node a is more influenced by nodes b and c than the other three nodes. However, since Kipf-GCN doesn't discriminate among the neighboring nodes, it is swayed by the majority and thereby estimating the wrong label L_1 for node a.

ConfGCN is able to overcome this problem by estimating confidences on each node's label scores. In Figure 1, such estimated confidences are shown by bars, with white and black bars denoting confidences in scores of labels L_0 and L_1, respectively. ConfGCN uses these label confidences to subdue nodes d, e, f since they have low confidence for their label L_1 (shorter black bars), whereas nodes b and c are highly confident about their labels being L_0 (taller white bars). This leads to higher influence of b and c during aggregation, and thereby ConfGCN correctly predicting the true label of node a as L_0 with high confidence. This clearly demonstrates the benefit of label confidences and their utility in estimating node influences. Graph Attention Networks (GAT) (Veličković et al., 2018), a recently proposed method also provides a mechanism to estimate influences by allowing nodes to attend to their neighborhood. However, as we shall see in Section 6, ConfGCN, through its use of label confidences, is considerably more effective.

Our contributions in this paper are as follows.

- We propose ConfGCN, a Graph Convolutional Network (GCN) framework for semi-supervised learning which models label distribution and their confidences for each node in the graph. To the best of our knowledge, this is the first confidence-enabled formulation of GCNs.
- ConfGCN utilize label confidences to estimate influence of one node on another in a label-specific manner during neighborhood aggregation of GCN learning.
- Through extensive evaluation on multiple real-world datasets, we demonstrate ConfGCN effectiveness over state-of-the-art baselines.

ConfGCN's source code and datasets used in the paper are available at `http://github.com/malllabiisc/ConfGCN`.

2 Related Work

Semi-Supervised learning (SSL) on graphs: SSL on graphs is the problem of classifying nodes in a graph, where labels are available only for a small fraction of nodes. Conventionally, the graph structure is imposed by adding an explicit graph-based regularization term in the loss function (Zhu et al., 2003; Weston et al., 2008; Belkin et al., 2006). Recently, implicit graph regularization via learned node representation has proven to be more effective. This can be done either sequentially or in an end to end fashion. Methods like DeepWalk (Perozzi et al., 2014), node2vec (Grover and Leskovec, 2016), and LINE (Tang et al., 2015) first learn graph representations via sampled random walk on the graph or breadth first search traversal and then use the learned representation for node classification. On the contrary, Planetoid (Yang et al., 2016) learns node embedding by jointly predicting the class labels and the neighborhood context in the graph. Recently, Kipf and Welling (2016) employs Graph Convolutional Networks (GCNs) to learn node representations.

Graph Convolutional Networks (GCNs): The generalization of Convolutional Neural Networks to non-euclidean domains is proposed by Bruna et al. (2013) which formulates the spectral and spatial construction of GCNs. This is later improved through an efficient localized filter approximation (Defferrard et al., 2016). Kipf and Welling (2016) provide a first-order

[2] In this paper, unless otherwise stated, we refer to Kipf-GCN whenever we mention GCN.

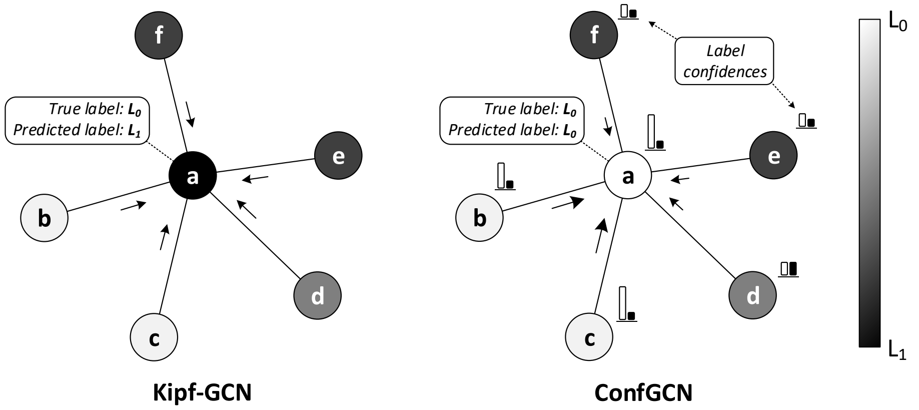


Figure 1: Label prediction on node a by Kipf-GCN and ConfGCN (this paper). L_0 is a's true label. Shade intensity of a node reflects the estimated score of label L_1 assigned to that node. Since Kipf-GCN is not capable of estimating influence of one node on another, it is misled by the dominant label L_1 in node a's neighborhood and thereby making the wrong assignment. ConfGCN, on the other hand, estimates confidences (shown by bars) over the label scores, and uses them to increase influence of nodes b and c to estimate the right label on a. Please see Section 1 for details.

formulation of GCNs and show its effectiveness for SSL on graphs. Marcheggiani and Titov (2017) propose GCNs for directed graphs and provide a mechanism for edge-wise gating to discard noisy edges during aggregation. This is further improved by Veličković et al. (2018) which allows nodes to attend to their neighboring nodes, implicitly providing different weights to different nodes. Liao et al. (2018) propose Graph Partition Neural Network (GPNN), an extension of GCNs to learn node representations on large graphs. GPNN first partitions the graph into subgraphs and then alternates between locally and globally propagating information across subgraphs. Recently, Lovasz Convolutional Networks Yadav et al. (2019) is proposed for incorporating global graph properties in GCNs. An extensive survey of GCNs and their applications can be found in Bronstein et al. (2017).

Confidence Based Methods: The natural idea of incorporating confidence in predictions has been explored by Li and Sethi (2006) for the task of active learning. Lei (2014) proposes a confidence based framework for classification problems, where the classifier consists of two regions in the predictor space, one for confident classifications and other for ambiguous ones. In representation learning, uncertainty (inverse of confidence) is first utilized for word embeddings by Vilnis and McCallum (2014). Athiwaratkun and Wilson (2018) further extend this idea to learn hierarchical word representation through encapsulation of probability distributions. Orbach and Crammer (2012) propose TACO (Transduction Algorithm with COnfidence), the first graph based method which learns label distribution along with its uncertainty for semi-supervised node classification. Bojchevski and Günnemann (2018) embeds graph nodes as Gaussian distribution using ranking based framework which allows to capture uncertainty of representation. They update node embeddings to maintain neighborhood ordering, i.e. 1-hop neighbors are more similar to 2-hop neighbors and so on. Gaussian embeddings have been used for collaborative filtering (Dos Santos et al., 2017) and topic modelling (Das et al., 2015) as well.

3 Notation & Problem Statement

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$ be an undirected graph, where $\mathcal{V} = \mathcal{V}_l \cup \mathcal{V}_u$ is the union of labeled ($\mathcal{V}_l$) and unlabeled ($\mathcal{V}_u$) nodes in the graph with cardinalities n_l and n_u, $\mathcal{E}$ is the set of edges and $\mathcal{X} \in \mathbb{R}^{(n_l+n_u)\times d}$ is the input node features. The actual label of a node v is denoted by a one-hot vector $Y_v \in \mathbb{R}^m$, where m is the number of classes. Given $\mathcal{G}$ and seed labels $Y \in \mathbb{R}^{n_l \times m}$, the goal is to predict the labels of the unlabeled nodes. To incorporate confidence, we additionally estimate label distribution $\boldsymbol{\mu}_v \in \mathbb{R}^m$ and a diagonal co-variance matrix $\boldsymbol{\Sigma}_v \in \mathbb{R}^{m \times m}$, $\forall v \in \mathcal{V}$. Here, $\boldsymbol{\mu}_{v,i}$ denotes the score of label i on node v, while $(\boldsymbol{\Sigma}_v)_{ii}$ denotes the variance in the estimation of $\boldsymbol{\mu}_{v,i}$. In other words, $(\boldsymbol{\Sigma}_v^{-1})_{ii}$ is ConfGCN's confidence in $\boldsymbol{\mu}_{v,i}$.

4 Background: Graph Convolutional Networks

In this section, we give a brief overview of Graph Convolutional Networks (GCNs) for undirected graphs as proposed by Kipf and Welling (2016). Given a graph $G = (\mathcal{V}, \mathcal{E}, \mathcal{X})$ as defined Section 3, the node representation after a single layer of GCN can be defined as

$$\boldsymbol{H} = f((\tilde{\boldsymbol{D}}^{-\frac{1}{2}}(\boldsymbol{A}+\boldsymbol{I})\tilde{\boldsymbol{D}}^{-\frac{1}{2}})\mathcal{X}\boldsymbol{W}) \quad (1)$$

where, $\boldsymbol{W} \in \mathbb{R}^{d \times d}$ denotes the model parameters, $\boldsymbol{A}$ is the adjacency matrix and $\tilde{\boldsymbol{D}}_{ii} = \sum_j (\boldsymbol{A}+\boldsymbol{I})_{ij}$. f is any activation function, we have used ReLU, $f(x) = \max(0, x)$ in this paper. Equation 1 can also be written as

$$\boldsymbol{h}_v = f\left(\sum_{u \in \mathcal{N}(v)} \boldsymbol{W}\boldsymbol{h}_u + \boldsymbol{b}\right), \quad \forall v \in \mathcal{V}. \quad (2)$$

Here, $\boldsymbol{b} \in \mathbb{R}^d$ denotes bias, $\mathcal{N}(v)$ corresponds to immediate neighbors of v in graph $\mathcal{G}$ including v itself and $\boldsymbol{h}_v$ is the obtained representation of node v.

For capturing multi-hop dependencies between nodes, multiple GCN layers can be stacked on top of one another. The representation of node v after k^{th} layer of GCN is given as

$$\boldsymbol{h}_v^{k+1} = f\left(\sum_{u \in \mathcal{N}(v)} \left(\boldsymbol{W}^k\boldsymbol{h}_u^k + \boldsymbol{b}^k\right)\right), \forall v \in \mathcal{V}. \quad (3)$$

where, $\boldsymbol{W}^k, \boldsymbol{b}^k$ denote the layer specific parameters of GCN.

5 Confidence Based Graph Convolution (ConfGCN)

Following (Orbach and Crammer, 2012), ConfGCN uses co-variance matrix based symmetric Mahalanobis distance for defining distance between two nodes in the graph. Formally, for any two given nodes u and v, with label distributions $\boldsymbol{\mu}_u$ and $\boldsymbol{\mu}_v$ and co-variance matrices $\boldsymbol{\Sigma}_u$ and $\boldsymbol{\Sigma}_v$, distance between them is defined as follows.

$$d_M(u, v) = (\boldsymbol{\mu}_u - \boldsymbol{\mu}_v)^T(\boldsymbol{\Sigma}_u^{-1} + \boldsymbol{\Sigma}_v^{-1})(\boldsymbol{\mu}_u - \boldsymbol{\mu}_v).$$

Characteristic of the above distance metric is that if either of $\boldsymbol{\Sigma}_u$ or $\boldsymbol{\Sigma}_v$ has large eigenvalues, then the distance will be low irrespective of the closeness of $\boldsymbol{\mu}_u$ and $\boldsymbol{\mu}_v$. On the other hand, if $\boldsymbol{\Sigma}_u$ and $\boldsymbol{\Sigma}_v$ both have low eigenvalues, then it requires $\boldsymbol{\mu}_u$ and $\boldsymbol{\mu}_v$ to be close for their distance to be low. Given the above properties, we define r_{uv}, the influence score of node u on its neighboring node v during GCN aggregation, as follows.

$$r_{uv} = \frac{1}{d_M(u, v)}.$$

This influence score gives more relevance to neighboring nodes with highly confident similar label, while reducing importance of nodes with low confident label scores. This results in ConfGCN acquiring anisotropic capability during neighborhood aggregation. For a node v, ConfGCN's equation for updating embedding at the k-th layer is thus defined as follows.

$$\boldsymbol{h}_v^{k+1} = f\left(\sum_{u \in \mathcal{N}(v)} r_{uv} \times \left(\boldsymbol{W}^k\boldsymbol{h}_u^k + \boldsymbol{b}^k\right)\right), \forall v \in \mathcal{V}. \quad (4)$$

The final node representation obtained from ConfGCN is used for predicting labels of the nodes in the graph as follows.

$$\hat{\boldsymbol{Y}}_v = \text{softmax}(\boldsymbol{W}^K\boldsymbol{h}_v^K + \boldsymbol{b}^K), \ \forall v \in \mathcal{V}$$

where, K denotes the number of ConfGCN's layers. Finally, in order to learn label scores $\{\boldsymbol{\mu}_v\}$ and co-variance matrices $\{\boldsymbol{\Sigma}_v\}$ jointly with other parameters $\{\boldsymbol{W}^k, \boldsymbol{b}^k\}$, following Orbach and Crammer (2012), we include the following two terms in ConfGCN's objective function.

For enforcing neighboring nodes to be close to each other, we include L_{smooth} defined as

$$L_{\text{smooth}} = \sum_{(u,v) \in \mathcal{E}} (\boldsymbol{\mu}_u - \boldsymbol{\mu}_v)^T(\boldsymbol{\Sigma}_u^{-1} + \boldsymbol{\Sigma}_v^{-1})(\boldsymbol{\mu}_u - \boldsymbol{\mu}_v).$$

To impose the desirable property that the label distribution of nodes in $\mathcal{V}_l$ should be close to their input label distribution, we incorporate L_{label} defined as

$$L_{\text{label}} = \sum_{v \in \mathcal{V}_l} (\boldsymbol{\mu}_v - \boldsymbol{Y}_v)^T(\boldsymbol{\Sigma}_v^{-1} + \frac{1}{\gamma}\boldsymbol{I})(\boldsymbol{\mu}_v - \boldsymbol{Y}_v).$$

Here, for input labels, we assume a fixed uncertainty of $\frac{1}{\gamma}\boldsymbol{I} \in \mathbb{R}^{L \times L}$, where $\gamma > 0$. We also include the following regularization term, L_{reg} to constraint the co-variance matrix to be finite and positive.

$$L_{\text{reg}} = \sum_{v \in \mathcal{V}} \text{Tr} \max(-\boldsymbol{\Sigma}_v, 0),$$

This regularization term enforces soft positivity constraint on co-variance matrix. Additionally in ConfGCN, we include the L_{const} in the objective, to push the label distribution ($\boldsymbol{\mu}$) close to the final model prediction ($\hat{\boldsymbol{Y}}$).

$$L_{\text{const}} = \sum_{v \in \mathcal{V}} (\boldsymbol{\mu}_v - \hat{\boldsymbol{Y}}_v)^T(\boldsymbol{\mu}_v - \hat{\boldsymbol{Y}}_v).$$

Dataset	Nodes	Edges	Classes	Features	Label Mismatch	$\frac{\lvert\mathcal{V}_l\rvert}{\lvert\mathcal{V}\rvert}$
Cora	2,708	5,429	7	1,433	0.002	0.052
Cora-ML	2,995	8,416	7	2,879	0.018	0.166
Citeseer	3,327	4,372	6	3,703	0.003	0.036
Pubmed	19,717	44,338	3	500	0.0	0.003

Table 1: Details of the datasets used in the paper. Please refer Section 6.1 for more details.

Finally, we include the standard cross-entropy loss for semi-supervised multi-class classification over all the labeled nodes ($\mathcal{V}_l$).

$$L_{\text{cross}} = -\sum_{v\in\mathcal{V}_l}\sum_{j=1}^{m} \boldsymbol{Y}_{vj}\log(\hat{\boldsymbol{Y}}_{vj}).$$

The final objective for optimization is the linear combination of the above defined terms.

$$\begin{aligned}
L(\theta) = &-\sum_{i\in\mathcal{V}_l}\sum_{j=1}^{L} \boldsymbol{Y}_{ij}\log(\hat{\boldsymbol{Y}}_{ij}) \\
&+ \lambda_1 \sum_{(u,v)\in\mathcal{E}} (\boldsymbol{\mu}_u - \boldsymbol{\mu}_v)^T(\boldsymbol{\Sigma}_u^{-1} + \boldsymbol{\Sigma}_v^{-1})(\boldsymbol{\mu}_u - \boldsymbol{\mu}_v) \\
&+ \lambda_2 \sum_{u\in\mathcal{V}_l} (\boldsymbol{\mu}_u - \boldsymbol{Y}_u)^T(\boldsymbol{\Sigma}_u^{-1} + \frac{1}{\gamma}\boldsymbol{I})(\boldsymbol{\mu}_u - \boldsymbol{Y}_u) \\
&+ \lambda_3 \sum_{v\in\mathcal{V}} (\boldsymbol{\mu}_u - \hat{\boldsymbol{Y}}_u)^T(\boldsymbol{\mu}_u - \hat{\boldsymbol{Y}}_u) \\
&+ \lambda_4 \sum_{v\in\mathcal{V}} \text{Tr}\ \max(-\boldsymbol{\Sigma}_v, 0)
\end{aligned}$$

where, $\theta = \{\boldsymbol{W}^k, \boldsymbol{b}^k, \boldsymbol{\mu}_v, \boldsymbol{\Sigma}_v\}$ and $\lambda_i \in \mathbb{R}$, are the weights of the terms in the objective. We optimize $L(\theta)$ using stochastic gradient descent. We hypothesize that all the terms help in improving ConfGCN's performance and we validate this in Section 7.4.

6 Experiments

6.1 Datasets

For evaluating the effectiveness of ConfGCN, we evaluate on several semi-supervised classification benchmarks. Following the experimental setup of (Kipf and Welling, 2016; Liao et al., 2018), we evaluate on Cora, Citeseer, and Pubmed datasets (Sen et al., 2008). The dataset statistics is summarized in Table 1. Label mismatch denotes the fraction of edges between nodes with different labels in the training data. The benchmark datasets commonly used for semi-supervised classification task have substantially low label mismatch rate. In order to examine models on datasets with more heterogeneous neighborhoods, we also evaluate on Cora-ML dataset (Bojchevski and Günnemann, 2018).

All the four datasets are citation networks, where each document is represented using bag-of-words features in the graph with undirected citation links between documents. The goal is to classify documents into one of the predefined classes. We use the data splits used by (Yang et al., 2016) and follow similar setup for Cora-ML dataset. Following (Kipf and Welling, 2016), additional 500 labeled nodes are used for hyperparameter tuning.

Hyperparameters: We use the same data splits as described in (Yang et al., 2016), with a test set of 1000 labeled nodes for testing the prediction accuracy of ConfGCN and a validation set of 500 labeled nodes for optimizing the hyperparameters. The ranges of hyperparameters were adapted from previous literature (Orbach and Crammer, 2012; Kipf and Welling, 2016). The model is trained using Adam (Kingma and Ba, 2014) with a learning rate of 0.01. The weight matrices along with μ are initialized using Xavier initialization (Glorot and Bengio, 2010) and $\boldsymbol{\Sigma}$ matrix is initialized with identity. To avoid numerical instability we model $\boldsymbol{\Sigma^{-1}}$ directly and compute $\boldsymbol{\Sigma}$ wherever required. Following Kipf and Welling (2016), we use two layers of GCN (K) for all the experiments in this paper.

6.2 Baselines

For evaluating ConfGCN, we compare against the following baselines:

- **Feat** (Yang et al., 2016) takes only node features as input and ignores the graph structure.
- **ManiReg** (Belkin et al., 2006) is a framework for providing data-dependent geometric regularization.
- **SemiEmb** (Weston et al., 2008) augments deep architectures with semi-supervised regularizers to improve their training.
- **LP** (Zhu et al., 2003) is an iterative iterative label propagation algorithm which propagates a nodes labels to its neighboring unlabeled nodes according to their proximity.

Method	Citeseer	Cora	Pubmed	Cora ML
LP (Zhu et al., 2003)	45.3	68.0	63.0	-
ManiReg (Belkin et al., 2006)	60.1	59.5	70.7	-
SemiEmb (Weston et al., 2008)	59.6	59.0	71.1	-
Feat (Yang et al., 2016)	57.2	57.4	69.8	-
DeepWalk (Perozzi et al., 2014)	43.2	67.2	65.3	-
GGNN (Li et al., 2015)	68.1	77.9	77.2	-
Planetoid (Yang et al., 2016)	64.9	75.7	75.7	-
Kipf-GCN (Kipf and Welling, 2016)	69.4 ± 0.4	80.9 ± 0.4	76.8 ± 0.2	85.7 ± 0.3
G-GCN (Marcheggiani and Titov, 2017)	69.6 ± 0.5	81.2 ± 0.4	77.0 ± 0.3	86.0 ± 0.2
GPNN (Liao et al., 2018)	68.1 ± 1.8	79.0 ± 1.7	73.6 ± 0.5	69.4 ± 2.3
GAT (Veličković et al., 2018)	72.5 ± 0.7	**83.0 ± 0.7**	79.0 ± 0.3	83.0 ± 0.8
ConfGCN (this paper)	**72.7 ± 0.8**	82.0 ± 0.3	**79.5 ± 0.5**	**86.5 ± 0.3**

Table 2: Performance comparison of several methods for semi-supervised node classification on multiple benchmark datasets. ConfGCN performs consistently better across all the datasets. Baseline method performances on Citeseer, Cora and Pubmed datasets are taken from Liao et al. (2018); Veličković et al. (2018). We consider only the top performing baseline methods on these datasets for evaluation on the Cora-ML dataset. Please refer Section 7.1 for details.

- **DeepWalk** (Perozzi et al., 2014) learns node features by treating random walks in a graph as the equivalent of sentences.
- **Planetoid** (Yang et al., 2016) provides a transductive and inductive framework for jointly predicting class label and neighborhood context of a node in the graph.
- **GCN** (Kipf and Welling, 2016) is a variant of convolutional neural networks used for semi-supervised learning on graph-structured data.
- **G-GCN** (Marcheggiani and Titov, 2017) is a variant of GCN with edge-wise gating to discard noisy edges during aggregation.
- **GGNN** (Li et al., 2015) is a generalization of RNN framework which can be used for graph-structured data.
- **GPNN** (Liao et al., 2018) is a graph partition based algorithm which propagates information after partitioning large graphs into smaller subgraphs.
- **GAT** (Veličković et al., 2018) is a graph attention based method which provides different weights to different nodes by allowing nodes to attend to their neighborhood.

7 Results

In this section, we attempt to answer the following questions:

Q1. How does ConfGCN compare against existing methods for the semi-supervised node classification task? (Section 7.1)

Q2. How do the performance of methods vary with increasing node degree and neighborhood label mismatch? (Section 7.2)

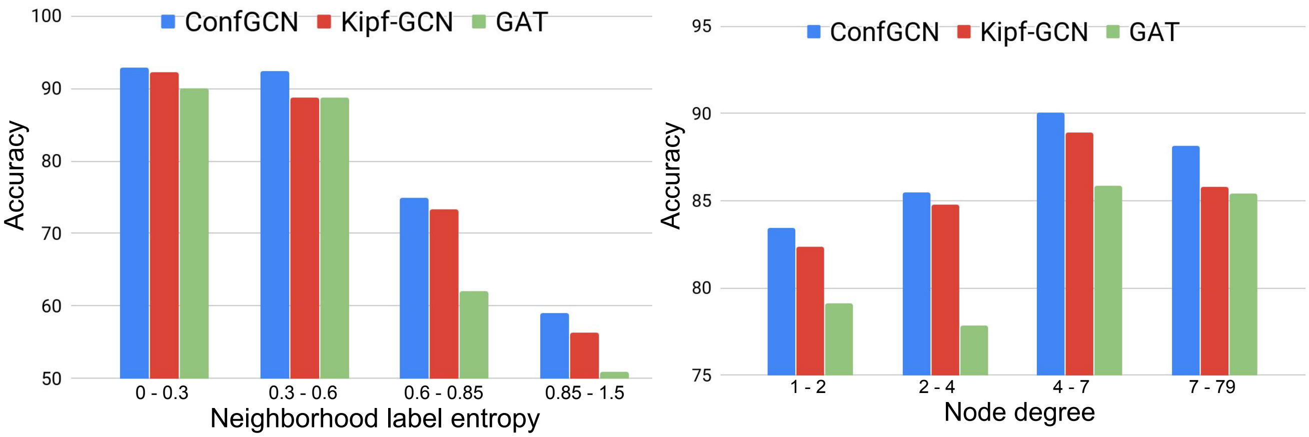
Q3. How does increasing the number of layers effect ConfGCN's performance? (Section 7.3)

Q4. What is the effect of ablating different terms in ConfGCN's loss function? (Section 7.4)

7.1 Node Classification

The evaluation results for semi-supervised node classification are summarized in Table 2. Results of all other baseline methods on Cora, Citeseer and Pubmed datasets are taken from (Liao et al., 2018; Veličković et al., 2018) directly. For evaluation on the Cora-ML dataset, only top performing baselines from the other three datasets are considered. Overall, we find that ConfGCN outperforms all existing approaches consistently across all the datasets.

This may be attributed to ConfGCN's ability to model nodes' label distribution along with the confidence scores which subdues the effect of noisy nodes during neighborhood aggregation. The lower performance of GAT (Veličković et al., 2018) compared to Kipf-GCN on Cora-ML shows that computing attention based on the hidden representation of nodes is not much helpful in suppressing noisy neighborhood nodes. We also observe that the performance of GPNN (Liao et al., 2018) suffers on the Cora-ML dataset. This is due to the fact that while propagating information between



(a)

(b)

Figure 2: Plots of node classification accuracy vs. (a) neighborhood label entropy and (b) node degree. On x-axis, we plot quartiles of (a) neighborhood label entropy and (b) degree, i.e., each bin has 25% of the samples in sorted order. Overall, we observe that ConfGCN performs better than Kipf-GCN and GAT at all levels of node entropy and degree. Please see Section 7.2 for details.

small subgraphs, the high label mismatch rate in Cora-ML (please see Table 1) leads to wrong information propagation. Hence, during the global propagation step, this error is further magnified.

7.2 Effect of Node Entropy and Degree on Performance

In this section, we provide an analysis of the performance of Kipf-GCN, GAT and ConfGCN for node classification on the Cora-ML dataset which has higher label mismatch rate. We use neighborhood label entropy to quantify label mismatch, which for a node u is defined as follows.

$$\text{NeighborLabelEntropy}(u) = -\sum_{l=1}^{L} p_{ul} \log p_{ul}$$

$$\text{where, } \; p_{ul} = \frac{|\{v \in \mathcal{N}(u) \mid \text{label}(v) = l\}|}{|\mathcal{N}(u)|}.$$

Here, label(v) is the true label of node v. The results for neighborhood label entropy and node degree are summarized in Figures 2a and 2b, respectively. On the x-axis of these figures, we plot quartiles of label entropy and degree, i.e., each bin has 25% of the instances in sorted order. With increasing neighborhood label entropy, the node classification task is expected to become more challenging. We indeed see this trend in Figure 2a where performances of all the methods degrade with increasing neighborhood label entropy. However, ConfGCN performs comparatively better than the existing state-of-art approaches at all levels of node entropy.

In case of node degree also (Figure 2b), we find that ConfGCN performs better than Kipf-GCN and GAT at all quartiles of node degrees. Classifying sparsely connected nodes (first and second bins) is challenging as very little information is present in the node neighborhood. Performance improves with availability of moderate number of neighboring nodes (third bin), but further increase in degree (fourth bin) results in introduction of many potentially noisy neighbors, thereby affecting performance of all the methods. For higher degree nodes, ConfGCN gives an improvement of around 3% over GAT and Kipf-GCN. This shows that ConfGCN, through its use of label confidences, is able to give higher influence score to relevant nodes in the neighborhood during aggregation while reducing importance of the noisy ones.

7.3 Effect of Increasing Convolutional Layers

Recently, Xu et al. (2018) highlighted an unusual behavior of Kipf-GCN where its performance degrades significantly with increasing number of layers. This is because of increase in the number of influencing nodes with increasing layers, resulting in “averaging out" of information during aggregation. For comparison, we evaluate the performance of Kipf-GCN and ConfGCN on citeseer dataset with increasing number of convolutional layers. The results are summarized in Figure 3. We observe that Kipf-GCN's performance degrades drastically with increasing number of layers, whereas ConfGCN's decrease in performance is more gradual. This shows that confidence based GCN helps in alleviating this problem. We also note that ConfGCN

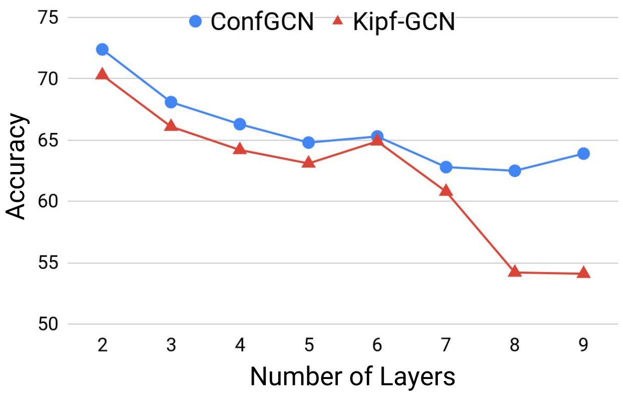


Figure 3: Evaluation of Kipf-GCN and ConfGCN on the citeseer dataset with increasing number of GCN layers. Overall, ConfGCN outperforms Kipf-GCN, and while both methods' performance degrade with increasing layers, ConfGCN's degradation is more gradual than Kipf-GCN's abrupt drop. Please see Section 7.3 for details.

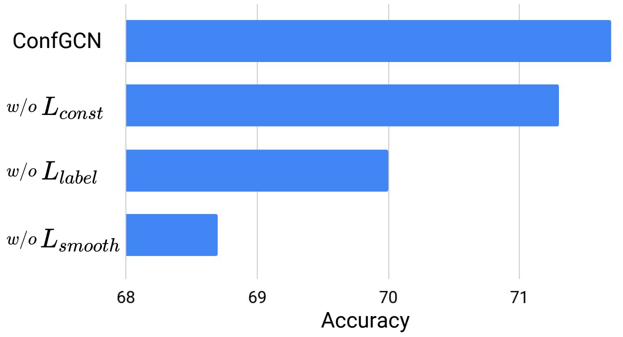


Figure 4: Performance comparison of different ablated version of ConfGCN on the citeseer dataset. These results justify inclusion of the different terms in ConfGCN's objective function. Please see Section 7.4 for details.

outperforms Kipf-GCN at all layer levels.

7.4 Ablation Results

In this section, we evaluate the different ablated version of ConfGCN by cumulatively eliminating terms from its objective function as defined in Section 5. The results on citeseer dataset are summarized in Figure 4. Overall, we find that each term ConfGCN's loss function (Equation 5) helps in improving its performance and the method performs best when all the terms are included.

8 Conclusion

In this paper, we present ConfGCN, a confidence based Graph Convolutional Network which estimates label scores along with their confidences jointly in a GCN-based setting. In ConfGCN, the influence of one node on another during aggregation is determined using the estimated confidences and label scores, thus inducing anisotropic behavior to GCN. We demonstrate the effectiveness of ConfGCN against state-of-the-art methods for the semi-supervised node classification task and analyze its performance in different settings. We make ConfGCN's source code available.

References

Athiwaratkun, B. and Wilson, A. G. (2018). On modeling hierarchical data via probabilistic order embeddings. In *International Conference on Learning Representations*.

Bastings, J., Titov, I., Aziz, W., Marcheggiani, D., and Simaan, K. (2017). Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967. Association for Computational Linguistics.

Belkin, M., Niyogi, P., and Sindhwani, V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.*, 7:2399–2434.

Bojchevski, A. and Günnemann, S. (2018). Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. In *International Conference on Learning Representations*.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42.

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. (2013). Spectral networks and locally connected networks on graphs. *CoRR*, abs/1312.6203.

Das, R., Zaheer, M., and Dyer, C. (2015). Gaussian lda for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 795–804. Association for Computational Linguistics.

Defferrard, M., Bresson, X., and Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. *CoRR*, abs/1606.09375.

Dos Santos, L., Piwowarski, B., and Gallinari, P. (2017). Gaussian embeddings for collaborative filtering. In

Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '17, pages 1065–1068, New York, NY, USA. ACM.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256.

Grover, A. and Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Huang, Z., Wan, C., Probst, T., and Van Gool, L. (2017). Deep learning on lie groups for skeleton-based action recognition. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1243–1252. IEEE computer Society.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kipf, T. N. and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907.

Lei, J. (2014). Classification with confidence. *Biometrika*, 101(4):755–769.

Li, M. and Sethi, I. K. (2006). Confidence-based active learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(8):1251–1261.

Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. (2015). Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*.

Liao, R., Brockschmidt, M., Tarlow, D., Gaunt, A., Urtasun, R., and Zemel, R. S. (2018). Graph partition neural networks for semi-supervised classification.

Marcheggiani, D. and Titov, I. (2017). Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515. Association for Computational Linguistics.

Orbach, M. and Crammer, K. (2012). Graph-based transduction with confidence. In *ECML/PKDD*.

Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 701–710, New York, NY, USA. ACM.

Sen, P., Namata, G. M., Bilgic, M., Getoor, L., Gallagher, B., and Eliassi-Rad, T. (2008). Collective classification in network data. *AI Magazine*, 29(3):93–106.

Subramanya, A. and Talukdar, P. P. (2014). Graph-based semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 8(4):1–125.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. (2015). Line: Large-scale information network embedding. In *WWW*. ACM.

Vashishth, S., Dasgupta, S. S., Ray, S. N., and Talukdar, P. (2018a). Dating documents using graph convolution networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1605–1615. Association for Computational Linguistics.

Vashishth, S., Joshi, R., Prayaga, S. S., Bhattacharyya, C., and Talukdar, P. (2018b). Reside: Improving distantly-supervised neural relation extraction using side information. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1257–1266. Association for Computational Linguistics.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph Attention Networks. *International Conference on Learning Representations*. accepted as poster.

Vilnis, L. and McCallum, A. (2014). Word representations via gaussian embedding. *CoRR*, abs/1412.6623.

Weston, J., Ratle, F., and Collobert, R. (2008). Deep learning via semi-supervised embedding. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1168–1175, New York, NY, USA. ACM.

Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.-i., and Jegelka, S. (2018). Representation learning on graphs with jumping knowledge networks. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5453–5462, Stockholmsmässan, Stockholm Sweden. PMLR.

Yadav, P., Nimishakavi, M., Yadati, N., Rajkumar, A., and Talukdar, P. (2019). Lovasz convolutional networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)"*.

Yang, Z., Cohen, W. W., and Salakhutdinov, R. (2016). Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 40–48. JMLR.org.

Yi, L., Su, H., Guo, X., and Guibas, L. (2016). Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. *arXiv preprint arXiv:1612.00606*.

Zhang, Y., Qi, P., and Manning, C. D. (2018). Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the Conference of Empirical Methods in Natural Language Processing (EMNLP '18)*, Brussels, Belgium.

Zhu, X., Ghahramani, Z., and Lafferty, J. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, pages 912–919. AAAI Press.