NODE CLASSIFICATION IN SHIFTING LANDSCAPES: FAIRNESS AWARE DOMAIN ADAPTATION IN NETWORK DATA

By

Anna Joy Stephens

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science - Master of Science

2023

ABSTRACT

Domain adaptation is an important task that considers how to apply a model trained on a labeled source dataset to an unlabeled target dataset. This is becoming an increasingly relevant concern as many datasets do not contain a large number of reliable labels and must consider additional training data from other sources. The domain adaptation task is challenging as the distributions of the source and target datasets may not be fully aligned. Such discrepancy in data distribution is called concept drift. This thesis focuses on the problem of domain adaptation in node classification for network data, a task known as cross-network node classification (CNNC). Unlike approaches specific to independent and identically distributed (i.i.d.) data, CNNC is concerned with the additional challenges introduced by the link structure of the source and target networks and differences in their node distributions. Such differences may exacerbate unfairness in node classification, and lead to one protected group receiving a disproportionately large number of positive outcomes. Additionally, existing CNNC approaches are computationally expensive as they rely upon having access to both the entire source and target graphs at one time.

In this thesis I first present OTGCN, a method for performing domain adaptation between multiple disconnected networks for the task of node classification. OTGCN combines the powerful graph convolutional network (GCN) architecture along with techniques from optimal transport to design source node embeddings that are more aligned with nodes in the target graph. This allows us to train a more accurate node classifier within the target domain. I then present `FOCI`, an improvement to OTGCN which addresses fairness by implementing a novel optimal transport approach designed to directly target harmful link bias. Lastly I introduce `FastFOCI` and `SpFOCI`, two further enhancements to `FOCI`, which directly address performance and statistical parity, a popular group fairness measure. I demonstrate the effectiveness of each of these methods on several real-world datasets and discuss their strengths and weaknesses.

This thesis is dedicated to my great-grandmother, Dorothy Marusek.
Thank you for always encouraging my curiosity and education.
Rest in peace.

ACKNOWLEDGEMENTS

I would like to acknowledge and thank everyone who has contributed to this work and my well-being along the way. First I would like to acknowledge my advisors Dr. Pang-Ning Tan and Dr. Abdol-Hossein Esfahanian. I would not be here without their patient instruction and assistance. Thank you as well to my committee member, Dr. Jiayu Zhou.

Next I would like to thank the other members of the DMiner lab at MSU. Fransisco Santos especially, as my partner on network fairness projects, has been there every step of the way to learn, share, encourage, and commiserate. Other members of the lab during my time there include Dr. Tyler Wilson, Dr. Boyang Liu, Galib Asadullah, Bidhan Bashyal, Andrew Mcdonald, and Yue Deng, who have all shared insights, code, data, and advice.

I would also like to thank my personal supporters through this process. My wonderful mother has never failed to encourage me or watch my dogs on long days. My brothers and faithful game masters have been my best friends and most vocal supporters, even when I cancel game sessions to meet a deadline. Last but certainly not least, my incredible fiancè, Joel, has supported me during every step of this crazy adventure from proof reading my initial applications to the lunch he's making me as I write this (which I will also ask him to proof read).

Lastly I would like to acknowledge the grant that has supported me through my time at MSU. This material is based upon work supported by the NSF Program on Fairness in AI in collaboration with Amazon under grant IIS-1939368. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or Amazon.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ASD Autism Spectrum Disorder.

CNNC Cross-Network Node Classification.

fMRI Functional Magnetic Resonance Imaging.

GCN Graph Convolutional Network.

GNN Graph Neural Network.

MMD Maximum Mean Discrepancy.

NFT Node Feature Transformation.

OT Optimal Transport.

PPMI positive pointwise mutual information.

TSNE T-distributed Stochastic Neighbor Embedding.

VAE Variational Auto-Encoders.

CHAPTER 1: INTRODUCTION

1.1 Motivation

Within the modern data landscape a typical dataset is often thought of as a single table filled with rows of independent and identically distributed (i.i.d.) data points. However, realistic datasets often take much more complex forms such as images, which must consider the spatial relationships between pixels, or time series, which considers sequence of the data points. Another complex dataset is the network, which considers both individual data points and the relationships between them. A network or graph consists of a set of nodes which are connected to each other by edges. These edges introduce relationships between data points which can be useful for modeling a wide range of important factors. Graphs can be used for modeling social networks, traffic patterns, brain structures, molecules, and many other scenarios in which structures, locations, similarity, or relationships are of importance.

One of the most common tasks when working with graphs is *node classification*. The standard node classification problem is to classify some unlabeled nodes when given a graph with node features, edge structure, and some labeled nodes. However, while the node classification task is common, real world scenarios are often much more nuanced. In particular, often a dataset of interest may not include enough labeled training data and must supplement from another labeled dataset. This can lead to complications as the datasets may not always have identical distributions. For example, social networks may represent different regions where node classification is complicated by divergence in societal norms. Population graphs used to model medical conditions in patient populations may come from large research institutions or small rural clinics. These scenarios motivate the task of *cross-network node classification*.

In cross-network node classification (CNNC) you typically have two graphs: a labeled *source* graph and an unlabeled *target* graph. The goal is to correctly classify nodes in the target graph. The source and target graphs contain the same features, but have varying distributions. The difference between the two distributions is called *concept shift* and is a

major focus of this work.

1.2 Challenges

In this thesis I consider three major challenges with CNNC. The primary challenge to any CNNC approach is to perform domain adaptation in a manner which respects the graph structure. In addition, fairness and performance are two very practical challenges when implementing a method for CNNC.

Domain adaptation is generally the task of classifying unlabeled target data points by making use of target feature information and a labeled source dataset. As described in the CNNC problem, the source and target datasets are often separated by concept drift. Most previous CNNC methods implement some form of adversarial domain adaptation [20] [67] [75] [56]. However, adversarial domain adaptation is often difficult to train and may not be effective in cases with a large amount of concept drift [30]. In this work I instead choose to address the domain adaptation challenge by implementing an *optimal transport* domain adaptation method. Optimal transport (OT) is a technique for discovering the best way to map between two distinct probability distributions. This map can be used to transform data points from one domain to another and can serve as a powerful tool in domain adaptation. Importantly, OT can be used to map between datasets with a large amount of concept drift and can be implemented with very little data.

Fairness in artificial intelligence can be grouped into two main categories. *Individual fairness* considers whether an outcome is fair to individuals, and *group fairness* considers if an outcome is fair to a group as a whole. The focus of this work is group fairness, and one of the most popular group fairness metrics is *statistical parity*. Statistical parity states that each protected group (e.g., males and females) should have an equal probability of experiencing a positive outcome. However, due to the fairness-utility trade off improving statistical parity may degrade model utility or performance [53] [15]. The challenge in this work is to incorporate statistical parity in a manner which balances this trade-off.

For various reasons most CNNC approaches encounter scalability issues when tasked

with large graphs. While these issues manifest at various technical points they all stem back to the same root cause. These CNNC approaches assume access to the entire graph in each step. For example, ASN, a state-of-the-art CNNC approach, implements variational auto-encoders which require access to complete source and target graphs and then a loss function with aggregation steps that can quickly explode as graphs increase in size [75]. In this work I specifically design a CNNC approach which is capable of scaling to large graphs if needed.

1.3 Key Contributions

In this thesis I introduce several novel contributions to address the challenges described above.

- I develop OTGCN, a CNNC approach that relies upon optimal transport based domain adaptation.
- I introduce two new fair Sinkhorn distance measures, one supervised and one unsupervised, which can be used to mitigate unfairness in optimal transport.
- I develop `FOCI`, a CNNC approach that addresses fair domain adaptation by implementing a fair optimal transport algorithm.
- I develop `FastFOCI` and `SpFOCI`, two additional CNNC approaches that extend the work of `FOCI` to add more scalability and an alternative approach to fairness.
- I perform experiments to demonstrate the effectiveness of these new approaches on several real-world datasets.

1.4 Thesis Organization

The remainder of this thesis will be organized in the following manner. *Chapter 2* contains a broad overview of various areas of relevant literature. Areas of literature presented in this chapter are relevant to the complete body of work, and are supplemented in later chapters with related work sections specific to the chapter of interest. *Chapter 3* presents OTGCN, our initial CNNC framework, and then demonstrates it as an effective approach when working with fMRI data collected from several different collection sites. *Chapter 4* presents two new approaches to optimal transport which allow it to consider fairness directly

within the transportation process. I then use these new optimal transport algorithms to develop several improvements to OTGCN which allow it to consider fairness and improve performance. All new methods are demonstrated experimentally on datasets of various sizes.

1.5 Publications

Much of the work in this thesis has been adapted from the following papers:

- Anna Stephens, Francisco Santos, Pang-Ning Tan, Abdol-Hossein Esfahanian. "Population Graph Cross-Network Node Classification for Autism Detection Across Sample Groups". In Proceedings of the Data Mining in Biomedical Informatics and Healthcare workshop at IEEE International Conference on Data Mining (DMBIH@ICDM, 2023), Shanghi (2023)
- Anna Stephens, Francisco Santos, Pang-Ning Tan, Abdol-Hossein Esfahanian. "FOCI: Fair Cross-Network Node Classification via Optimal Transport." (under review), 2023

CHAPTER 2: LITERATURE REVIEW

In this chapter I review literature related to several key areas of this project. Specifically this chapter includes literature on node or graph representation learning, cross-network node classification (CNNC), optimal transport as it has been used with graph data structures, and fairness in neural networks. Some of these areas are much too large to be fully reviewed within the confines of this chapter, and in those cases I refer the reader to survey papers for a more thorough discussion of the topic.

2.1 Node Representation Learning

In current literature the most common method for constructing node representations or embeddings is via graph neural networks (GNNs). GNNs generally consider graph structure by using message passing techniques to combine a node's features with features from other nodes in its neighborhood. Notably, node representation learning is a large field and this section will only hit some of the highlights. For more information please consider a survey paper on the topic [76] [36].

Arguably the most well known and straightforward GNN is the Graph Convolutional Network (GCN)[37]. GCN uses a simple message passing technique to combine node features with features from all of its immediate neighbors in one step. GCN layers can then be stacked to expand to additional neighbors beyond one hop. Most other convolutional GNNs build off the GCN in some way. JKNet [68] and MixHop [2] are approaches for expanding convolutions beyond the immediate neighborhood. FastGCN [12] and simple graph convolution (SGC) [66] are methods for simplifying and speeding up GCN. GraphSAGE [28] is also similar to GCN, but does not require the model to evaluate the whole graph at one time; instead performing strategic graph sampling before sending portions of the graph to a GCN-like model.

While graph convolutional networks are perhaps the largest and well known family of GNNS, graph attention networks follow closely behind. The most well known graph attention networks is aptly named Graph Attention Network (GAT) [62]. GAT introduces

an attention mechanism by which the model can weight various neighboring nodes differently when combining features. GATv2 [8] re-orders the steps of GAT in order to provide a more expressive representation. Multi-hop Attention Graph Neural Network (MAGNA) [63] also operates similarly to GAT, but considers larger neighborhoods in each layer or step.

2.2 Cross Network Node Classification

One of the earliest works on CNNC is the CDNE algorithm [57]. CDNE uses stacked autoencoders to learn separate embeddings for the source and target networks and tackles domain adaptation by utilizing maximum mean discrepancy (MMD) loss between the two embeddings. ACDNE[56] was a follow up work to CDNE which improved upon embedding construction by designing separate methods for the node features and graph structure. Additionally, it used an adversarial discriminator to address domain adaptation instead of the MMD loss.

More recently AdaGCN[20], UDA-GCN[67], and ASN[75] have attempted to address the CNNC task. All three of these methods relied upon the GCN acchitecture [37] to generate the node embeddings. AdaGCN[20] addresses the domain adaptation issue with an adversarial discriminator. UDA-GCN[67] followed on the heels of AdaGCN by utilizing the PPMI matrix instead of an adjacency matrix along with an entropy loss to update the target GCN. Finally ASN[75] is the most recent of these works and combined concepts from both AdaGCN and UDA-GCN. ASN used variational auto-encoders (VAE) to help preserve information unique to the source and target graphs respectively. None of these methods address fairness concerns.

2.3 Optimal Transport with Graphs

There is a small body of work investigating OT within the context of graph data. Some of these works [60][1] have developed OT-inspired distance measures for graph data, while others employ OT for graph comparison or alignment tasks instead of node classificaiton. For example, GOT[13] uses a combination of traditional Wasserstein distance with the Gromov-Wasserstein to find a transport plan for the purposes of graph alignment. Zhang et al. [74]

employed OT to fine-tune large GNNs while OT-GNN[11] used OT to develop a prototype for prototype learning on whole graph classification. Unfortunately, none of these works consider using OT for CNNC nor addressing the issue of algorithmic bias.

2.4 Fairness in Neural Networks

Previously I mentioned that fairness metrics and definitions can be grouped into ones targeting group fairness and individual fairness. On the other hand, fairness mechanisms are typically grouped based on when they occur in the machine learning pipeline: pre-process, in-process, and post-process. This section focuses on some popular fairness mechanisms. Similar to node representation learning, fairness in neural networks is a large topic, and I recommend a survey paper for a more in-depth review of both fairness history, metrics, and mechanisms [53].

Pre-processing fairness approaches modify the training data before sending it to a model for training. One approach does this by using a k-nearest neighbor classify to strategically flip training labels so that a later model will be more generous towards specific protected groups [45]. Another approach implements representation learning techniques to design a more fair feature set [71].

In-processing approaches modify the model directly in some way during training in order to train a fair model more directly. This is often done by adding additional terms to the model's loss function [32] [7] [5]. Sometimes, however, these approaches can become more creative. For example, one method uses maximum mean discrepancy (MMD) to align distributions of various protected groups [54]. Another method uses variational auto-encoders and MMD to remove any dependence on the protected attribute [44].

Post-processing methods modify model output in some way to encourage fairness. A straighforward example of this is using different probability thresholds for different protected groups in a binary classification task [47]. More generally one can strategically flip the labels of some predicted outputs to meet some specific fairness criteria [29].

CHAPTER 3: CNNC ACROSS SAMPLE GROUPS

3.1 Introduction

The proliferation of graph-based data in various application domains has motivated the need to develop more advanced techniques based on deep learning to harness the network data from multiple sources in order to improve the performance of node classification algorithms. Current techniques would learn a feature embedding of the nodes from multiple graphs using their node features and link structure information. The learned feature representation is then presented to a fully-connected network layer to perform the final classification. However, despite the notable advances in graph neural networks, there are still numerous challenges that must be addressed in order for the techniques to be successfully deployed to solve real-world problems.

First, existing techniques often assume that the graphs share similar distributional properties, thus enabling us to apply a model trained on one graph, *a.k.a.* the *source graph*, to the nodes in another, *a.k.a.* the *target graph*. Unfortunately, such a scenario is too idealistic for the real world as one would likely encounter some form of distributional shift, where the training data only captures the essence of a particular graph but fails to account for some unforeseen differences in another graph. Such type of concept drift is illustrated in Fig. 3.1,

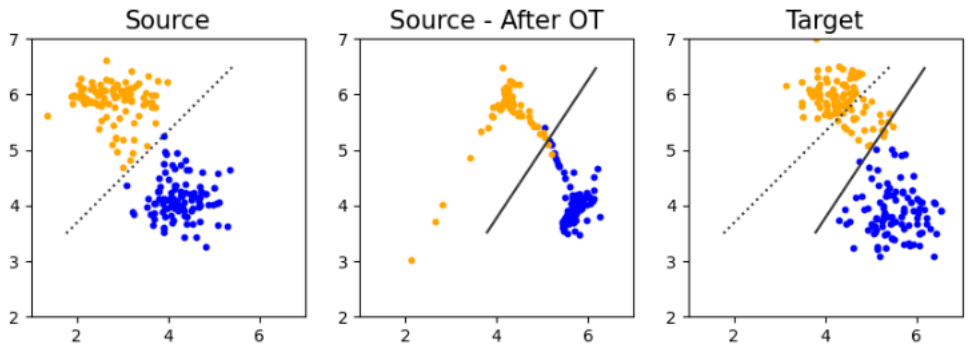


Figure 3.1: An illustration of OT for domain adaptation, where the color represents class labels. The diagram on the left is the original source dataset and on the far right is the target dataset. The dotted line is a the decision boundary of logistic regression trained on the source dataset. The middle diagram shows the transported source dataset using OT, with the solid line representing the decision boundary obtained by logistic regression.

where the decision boundary induced from the learned representation of a source graph (left) does not reflect the class separation of the nodes in the target graph (right).

To address the concept drift problem, domain adaptation and transfer learning methods [50] have been developed. Here, the deep neural network is initially trained using data from a related source domain. The model is then fine-tuned to fit characteristics of the target domain. Transfer learning is particularly useful when there is limited training data available in the target domain. For graph data, transfer learning is typically studied under the guise of *cross-network node classification* (CNNC) problems [56, 57, 20, 67, 75]. In CNNC, we consider a scenario in which there is a source graph, with fully or mostly labeled nodes, and a target graph, which has either a few or no class labels.

CNNC approaches are particularly well suited to address the multi-site Autism Spectrum Disorder (ASD) detection problem. ASD refers to a condition characterised by specific communication impairments, restricted interests, and repetitive behaviours [43]. As the disorder typically presents itself early in life, early and accurate detection can help reduce the severity of many lifelong symptoms. Towards this end, automated techniques based on machine learning and deep learning have been developed for the early detection of ASD. Specifically, these techniques have been applied to a wide range of subject data including detailed subject screening data [22], videos of subject movements [77], and Magnetic Resonance Imaging (MRI) data [42]. Given the diverse modalities of data available, it is likely that the best detection results can be achieved by combining the data from different sources.

Previous works have established an effective method to combine various modalities of data using graph structures because of their flexible yet powerful representation [61] [72]. These approaches make use of population graphs, where the nodes represent individuals and edges are generally defined with some similarity measure. In several prior ASD research, the node features are generated from image information while the edges consider a combination of subject information (sex, age, etc) in addition to image similarity [51] [52] [35] [31]. Once these graphs are constructed, detecting ASD becomes a *node classification* problem.

Similar to other diagnosis problems, one of the major challenges in applying machine learning to ASD detection is the limited amount of labeled data available. This has led to growing interest in utilizing labeled data from multiple sites to train the machine learning model. One limitation of these approaches is that they fail to consider a more realistic scenario in which there may be little or no labeled data associated with the population of interest. For example, a model may be trained on a research dataset but needs to be applied to another location with a different imaging equipment or patient demographics that were not well represented in the training data. This problem can be addressed using an approach generally known as domain adaptation or transfer learning [59]. Within the context of node classification tasks, it is also referred to as *cross-network node classification* (CNNC) approach. CNNC assumes the availability of a sufficiently large number of labeled nodes in a source network and an unlabeled target network, whose node labels are to be predicted accurately.

This chapter focuses on addressing the CNNC task for ASD detection. We use the popular ABIDE [17] dataset, which contains both resting-state functional Magnetic Resonance Imaging (fMRI) information and phenotypic data such as age, sex, and screening results. Samples in the ABIDE dataset were collected from several different collection sites, which were then divided into 2 groups to form the source and target networks for our experimental studies.

CNNC has two major challenges when applied to the ABIDE dataset. The first is extracting relevant information from the networks for ASD detection. In this work we will learn a graph embedding of the source and target networks via a graph convolutional network (GCN) [37]. A GCN layer is capable of learning a node embedding which contains information about a node and its immediate neighbors. This method relies on the homophily principle [46], which states that similar individuals tend to form neighborhoods within a graph. Therefore, in a graph with high homophily we can improve a node embedding by adding information from it's neighborhood.

The second major challenge is to handle potential concept drift between the source and target networks. As the separate data collection sites may have different fMRI imaging equipment and procedural differences, this may introduce some discrepancies or "drift" between the two networks. As previously mentioned, the presence of concept drift often leads to poor results if a classifier is simply trained on the source dataset and applied as it is to the target dataset. Fig. 3.1 demonstrates the challenge of CNNC when the learned node embedding does not account for such distributional shift. A decision boundary learned from the source dataset is likely to incorrectly classify a significant portion of the target dataset.

In this chapter we introduce OTGCN, a novel approach to address the CNNC task for ASD detection. Our proposed approach combines graph neural network with optimal transport (OT) to handle the drift between the source and target networks. OT is a method for mapping a transportation between two distributions. We will use OT to map the learned source representation to the target representation. This strategy allows us to train an accurate model for classifying the target nodes. Experimental results on the ABIDE dataset demonstrate the effectiveness of our approach at diagnosing ASD across different collection sites compared to state of the art CNNC methods.

3.2 Related Works

In previous chapters we have discussed works relevant to node classification and cross-network node classification tasks. In this chapter, however, we specifically focus on the use of optimal transport for addressing ASD detection across various sample collection sites. To that end, we must now explore works which are specifically relevant to detecting ASD as well as addressing fMRI data collected from multiple sites.

3.2.1 Machine Learning Approaches to ASD Detection

ASD detection is a task that lends itself to a wide variety of approaches. Zunino et al [77] employed computer vision approaches for automatic early detection of ASD by evaluating videos of subject movements when grasping a bottle. They then applied recurrent neural networks to distinguish subjects who have ASD from those who do not. Erkan and Thanh

[22] analyzed detailed screening data collected from mobile app surveys using traditional machine learning methods such as k-nearest neighbor, support vector machines, and random forests to diagnose subjects with ASD. Yuan et al. [70] applied natural language processing (NLP) techniques to analyze hand written medical forms of potential ASD patients while Carette et al. [10] performed ASD detection on eye tracking data using long-short term memory (LSTM) networks.

The majority of the recent works in this area, however, focuses on using MRI data [42] [49] along with other subject information such as sex, age, and screening results [51] [52] [31] [35] for screening potential ASD patients. For example, Li et al. [42] presented a graph neural network approach to find biomarkers that can be used to detect ASD while Parisot et al. [51] employed a graph convolutional network (GCN) to perform the detection using both fMRI imaging and non-imaging phenotypic data. Similar to these works, the work in this chapter focuses on using a combination of fMRI and other subject data, though our approach is also applicable to blend other forms of data given the representation power of graph neural networks.

3.2.2 Machine Learning on Multi-site fMRI Data

Previous works on diagnosing brain-related problems using fMRI data from multiple sites generally fall into two major categories—transfer learning and federated learning.

Transfer learning [59] is a machine learning approach that enables prediction models trained from a given data domain (known as the *source* domain) to be applied to another domain (known as the *target* domain). Such a domain adaptation approach can be used even if the target domain has no labeled training data. Previous works on applying transfer learning to fMRI data can be found in [14] [64] [27]. However, these approaches are mostly designed for using only the image information and do not consider more complex data structures or the use of additional non-imaging information.

In contrast, federated learning [73] is designed for training prediction models in a collaborative fashion on decentralized datasets. The approach assumes restricted or indirect

access to the source dataset and direct access to a partially labeled target dataset. There has been a few works applying federated learning approaches to multi-site fMRI data [41] [65], but none of them consider non-imaging information.

3.3 Preliminaries

3.3.1 Problem Statement

Let $\mathcal{G}(V, E, X, Y)$ be an attributed graph, with a node set V, edge set $E \subseteq V \times V$, node feature matrix X, and node label Y. Let A denote the adjacency matrix representation of E, where $A_{ij} > 0$ if $(v_i, v_j) \in E$ and 0 otherwise. In a domain adaptation setting, we assume there exists a source graph, $\mathcal{G}_s(V_s, E_s, X_s, Y_s)$, where the node labels in Y_s are known, and a target graph, $\mathcal{G}_t(V_t, E_t, X_t, Y_t)$, where the node labels in Y_t are unknown. The adjacency matrices corresponding to the source and target graphs are denoted as $A_s \in \mathbb{R}^{n_s \times n_s}$ and $A_t \in \mathbb{R}^{n_t \times n_t}$, respectively, where n_s and n_t are the corresponding number of source and target nodes.

We further assume that both graphs have identical features, i.e., $X_s \in \mathbb{R}^{n_s \times m}$ and $X_t \in \mathbb{R}^{n_t \times m}$. Both graphs are also assumed to have the same set of class labels, i.e., $Y_s \in \{0, 1, \cdots, k-1\}^{n_s}$ and $Y_t \in \{0, 1, \cdots, k-1\}^{n_t}$, where k is the number of classes. For ASD detection, we consider a binary node classification problem with $k = 2$. At times, we also consider the combined graph $\mathcal{G}_c(V_c, E_c, X_c, Y_c)$, where $V_c = V_s \cup V_t$, $E_c = E_s \cup E_t$, $X_c = [X_s; X_t] \in \mathbb{R}^{n \times m}$, $A_c = [A_s\ \mathbf{0};\ \mathbf{0}^T\ A_t] \in \mathbb{R}^{n \times n}$, $Y_c \in \{0, 1, \cdots, k-1\}^n$, $n = n_s + n_t$, and $\mathbf{0}$ is an $n_s \times n_t$ null matrix.

Definition 1 (Cross Network Node Classification (CNNC))**.** *Given a source graph, $\mathcal{G}_s = (V_s, E_s, X_s, Y_s)$ and target graph, $\mathcal{G}_t = (V_t, E_t, X_t, Y_t)$, our goal is to learn a target function, $f : V \rightarrow \{0, 1, \cdots, k-1\}$, that accurately classifies the labeled nodes in Y_s as well as the unlabeled nodes in Y_t.*

3.4 Graph Convolutional Network (GCN)

Graph convolutional networks [37] employ a message passing strategy to succinctly capture both the node features and graph structure information when learning the feature

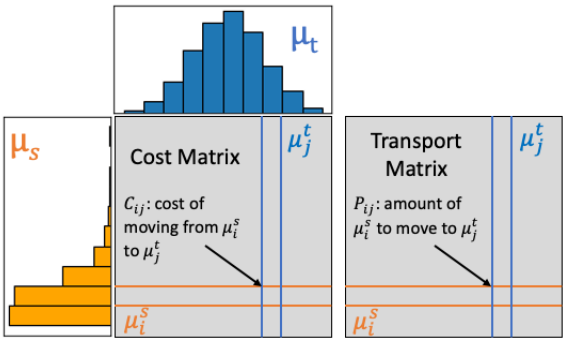


Figure 3.2: An illustration of the cost and transport plan matrices of optimal transport.

embedding of a node in the graph. Specifically, each "message" corresponds to the feature embedding information of a node, which will be passed to all of its immediate neighbors. By aggregating the features gathered from the neighbors, a new embedding for the node will be generated.

The message passing strategy can be formally stated as follows. Given an adjacency matrix A and node feature matrix X, the feature embedding of the nodes in layer $l+1$, $H^{(l+1)}$, can be computed based on its feature embedding at its previous layer, $H^{(l)}$, as follows:

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}) \tag{1}$$

$$\tilde{A} = A + I$$

$$\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$$

where $H^{(0)} = X$ (i.e., original node features), $\sigma(\cdot)$ is the ReLU activation function, and $\tilde{D}$ is a diagonal matrix containing the sum of the edge weights associated with each node in the graph.

3.5 Optimal Transport

Optimal Transport (OT) provides a principled approach for comparing two probability distributions by finding the least costly way to reshape one of the distributions into the other while incorporating their geometric information. The original OT problem was proposed by Monge [48], with its modern formulation being attributed to Kantorovich [33]. Given two marginal distributions $\mu_s \in \mathbb{R}^{n_s}$ and $\mu_t \in \mathbb{R}^{n_t}$ let $C \in \mathbb{R}^{n_s \times n_t}$ be a cost matrix, where C_{ij}

is the cost of transporting one unit from μ_i^s to μ_j^t. Consider a transportation plan matrix $P \in \mathbb{R}^{n_s \times n_t}$, where p_{ij} is the proportion of probability mass μ_i^s that should be moved to μ_j^t.

The optimal transport (OT) problem seeks to find a transportation plan P that minimizes the total transportation cost. The minimum transportation cost $W(\mu_s, \mu_t)$ is also called the Wasserstein distance or the earth-mover distance.

$$W(\mu_s, \mu_t) := \min_{P \in U(\mu_s, \mu_t)} \langle P, C \rangle_F \tag{2}$$
$$U(\mu_s, \mu_t) := \{P \in \mathbb{R}_+^{n_s \times n_t} | P1_{n_t} = \mu_s, P^T 1_{n_s} = \mu_t\}$$

The discrete OT formulation [33] shown above can be solved as a linear programming problem. However, it is computationally expensive $\mathcal{O}((n+m)nm\log(n+m))$, unstable, and is not guaranteed to have unique solution. Fortunately these problems can be addressed by employing the Sinkhorn distance W_λ [18] shown below, which utilizes an entropy regularization function $H(P)$ to accelerate the OT computation. Specifically, $W_\lambda(\mu_s, \mu_t)$ and P^λ can be solved using the well-known Sinkhorn algorithm as established by Cuturi [18].

$$\begin{aligned} W_\lambda(\mu_s, \mu_t) &= \langle P^\lambda, C \rangle \\ P^\lambda &= \operatorname*{argmin}_{P \in U(\mu_s, \mu_t)} \langle P, C \rangle - \lambda H(P) \\ U(\mu_s, \mu_t) &:= \left\{ P \in \mathbb{R}_+^{n_s \times n_t} \middle| P1_{n_t} = \mu_s, P^T 1_{n_s} = \mu_t \right\} \end{aligned} \tag{3}$$

where $H(P) = -\sum_{ij} P_{ij} \log P_{ij}$ is the entropy regularization and λ is a user-specified regularization hyperparameter.

3.6 Methodology

We employed a combination of graph convolutional networks (GCN) with node feature transformation layers to learn the feature embedding of the nodes in the source and target graphs. We then performed optimal transport on the learned embedding of the source nodes to match the distribution of the learned embedding of the target nodes. Figure 3.3 provides a high-level illustration of our proposed deep neural network architecture along with its

training procedure. Details of the proposed architecture are described in the subsections below.

3.6.1 Initial Node Embedding Construction & Pretraining

As previously noted, conventional GCN employs a message passing strategy to transmit information about the feature embedding of a node to its immediate neighbors. This allows each node to aggregate the feature information of its neighbors when constructing its own latent embedding. Unfortunately, in graphs with low homophily, it is possible that the neighborhood information obtained via message passing is of less value than the original feature information of the node itself, which was the case with the population graph constructed

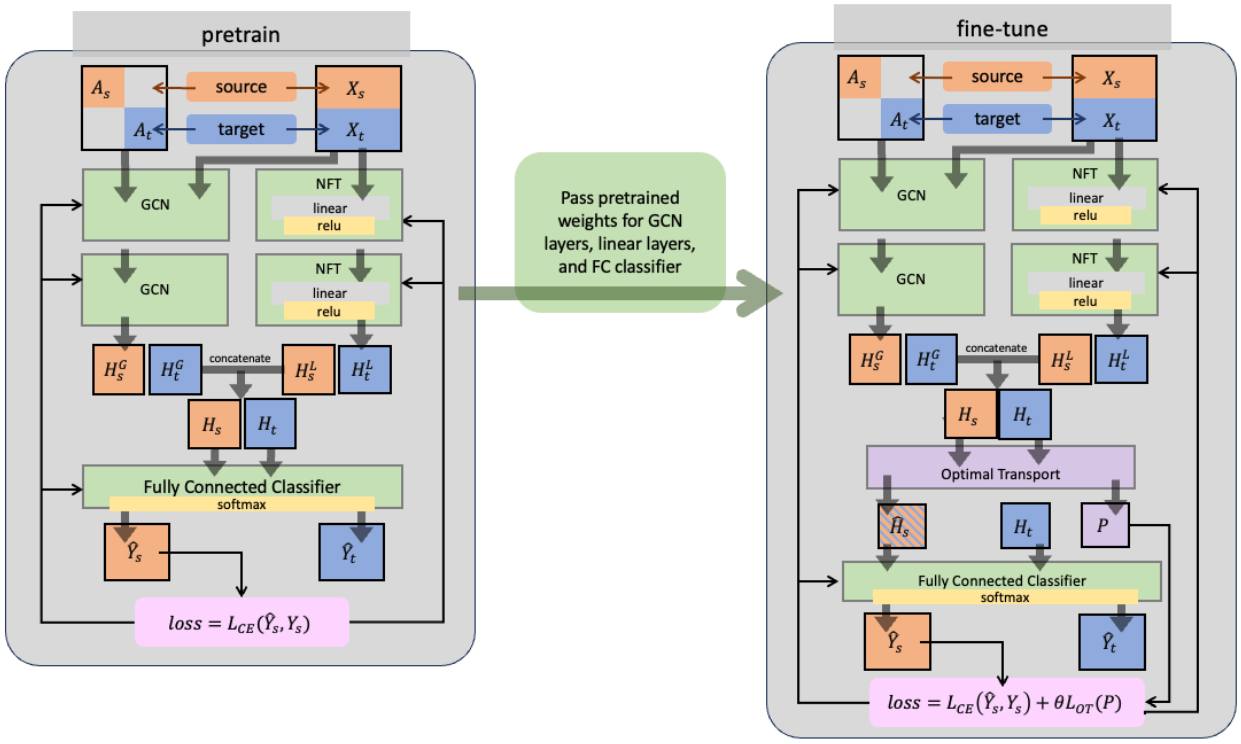


Figure 3.3: High level architecture of our model. Combined source and target datasets are fed to two GCN layers and two NFT layers to learn two distinct embeddings. In pretraining (left) these embeddings are concatenated and sent directly to a fully connected classifier. Model weights are then updated using just cross entropy loss. In OT training (right) concatenated embeddings are routed to an OT layer before being sent to the classifier. The OT layer replaces the source embedding with a transported version and then sends the target embedding and the transported source embedding to the classifier. At this point the model weights are updated with a combination of cross entropy and OT losses.

from the ABIDE dataset. However, due to its current formulation (see Eqn. 1), conventional GCN may not be able to attenuate the influence of the graph structure in comparison to the influence of the original features.

To overcome this challenge, we propose a modification to the graph convolutional network architecture to independently learn a nonlinear embedding of the original node features. We call this the *node feature transformation* (NFT) layer in Fig 3.3. The NFT layer consists of a combination of linear layer plus ReLU activation functions to transform the original node features into their corresponding nonlinear embedding. The transformed features are then concatenated with the structural embedding of the nodes obtained from GCN for subsequent node classification. This strategy increases the flexibility of the model to capture the relative influence of the node features and homophily (i.e., neighborhood features) on the classification task. As shown in the architecture diagram, the GCN layers are trained on combined adjacency and feature matrices of the source and target graphs, while the NFT layers are trained on the combined feature matrices. The weights of the GCN and NFT layers are jointly updated during backpropagation.

The pretraining process of the deep neural network can be seen on the left side of Fig 3.3. The purpose of pretraining is to ensure that the full network with optimal transport (right side of Fig 3.3) can be seeded with a good set of initial weights. During pretraining, we train the NFT and GCN layers to each produce their own feature embeddings, H^L (output of NFT layer) and H^G (output of GCN layer). The two embeddings are then concatenated and sent to a fully connected network for node classification. The network is trained to minimize the following cross-entropy loss function:

$$\text{Pretraining:} \quad \mathcal{L}_{CE} = -\frac{1}{n_s} \sum_{x_i \in X_s} \sum_{j=0}^{l} Y_{ij} \log(\hat{Y}_{ij}) \tag{4}$$

3.6.2 Domain Adaptation via Optimal Transport GCN

We address the domain adaptation problem for cross network node classification (CNNC) by combining the pretrained network with an optimal transport layer as shown on the right

side of Fig. 3.3. The optimal transport layer utilizes the Sinkhorn algorithm [18] to learn the relevant transportation plan matrix P that will transform the learned embedding of the source nodes to match the distribution of the target node embedding.

The OT layer takes the source and target node embeddings along with the entropy regularization hyperparameter λ as inputs and returns a new embedding of the source nodes, $\hat{H}_s$, which matches the distribution of the target nodes, H_t. This can be accomplished by solving the OT problem given in Eqn. (3) to learn the transportation plan P and using the barycentric mapping approach in [16] [23] to transform the source node features. Specifically, for each source node i, its latent features will be transported to a new embedding as follows:

$$\hat{H}_{s,i} = \underset{H \in \mathbb{R}^d}{\operatorname{argmin}} \sum_j P^{\lambda}_{i,j} \; C(H, H^t_j)$$

If the cost function C corresponds to the squared l_2 distance, then the barycentric mapping reduces to the following form.

$$\hat{H}_s = \operatorname{diag}(P^{\lambda} 1_{n_t})^{-1} P^{\lambda} H_t$$

For domain adaptation, we typically choose the marginal distributions of the source and target node embeddings, μ_s and μ_t, to be a uniform distribution. This allows us to further simplify the equation as follows:

$$\hat{H}_s = n_s P^{\lambda} H_t \tag{5}$$

where n_s is the number of source nodes.

Both $\hat{H}_s$ and H_t are then fed to a fully connected layer to perform the node classification. By replacing H_s with $\hat{H}_s$ we train the model to work on data that has been transported to the target domain. This addresses the domain adaptation problem and allows the classifier to accurately classify the target nodes.

Finally, the full OTGCN network is trained to minimize the following joint objective function:

$$\mathcal{L} = \mathcal{L}_{CE} + \theta \mathcal{L}_{OT} \tag{6}$$

where $\mathcal{L}_{CE}$ is the cross entropy loss given in Eqn. (8) and $\mathcal{L}_{OT}$ is the optimal transport loss defined as follows:

$$\mathcal{L}_{OT} = \langle P^{\lambda}, C \rangle - \lambda H(P^{\lambda}) \tag{7}$$

with the hyperparameter θ controlling how much emphasis is placed on the optimal transport term.

3.7 Experimental Evaluation

This section presents the experiments performed to evaluate the effectiveness of our proposed OTGCN framework. The source code for the framework and other aspects of our experiments can be found at https://github.com/ajoystephens/otgcn

Site	**Samples**	**Autism**	**Dataset**
YALE	41	22	target
CALTECH	15	5	target
CMU	11	6	target
NYU	172	74	source
UM_1	86	34	source
USM	67	43	source
UCLA_1	64	37	source
PITT	50	24	source
MAX_MUN	46	19	source
TRINITY	44	19	source
UM_2	34	13	source
KKI	33	12	source
LEUVEN_2	28	12	source
LEUVEN_1	28	14	source
OLIN	28	14	source
SDSU	27	8	source
SBL	26	12	source
STANFORD	25	12	source
OHSU	25	12	source
UCLA_2	21	11	source

Table 3.1: The data collection sites for ABIDE and its selection as source/target graph.

	Autism	Control	Total
Source	370 (46.0%)	434 (54.0%)	804
Target	33 (49.3%)	34 (50.7%)	67

Table 3.2: ABIDE Totals by Label and Dataset.

3.7.1 Data Preparation

The ABIDE [17] dataset is a combination of samples obtained from 20 different collection sites as shown in Table 4.1 for ASD classification. We split these collection sites into source and target sites for a total of 804 source samples and 67 target samples. Following the terminology used in other previous works in this area [51] [64] [42] we refer to the two classes in the dataset as *autism* and *control*. Autism refers to a subject who developed ASD while control refers to a subject without ASD.

For the ABIDE dataset, the image data was prepared following the methodology of [51]. A population graph is constructed by considering each subject as a node and computing the similarity of their fMRI images for edge construction. Specifically, an edge was placed between a pair of nodes only if their image similarity exceeds certain threshold value. The edges are also weighted according to their computed similarity. The node features were derived from phenotype information. After removing all columns with a known direct relationship to the class label, the features were one-hot-encoded as applicable. They were then normalized and any features with an abnormally high correlation to the label were also removed.

3.7.2 Experimental Setup

We compared the performance of OTGCN against the following baselines. Aside from GCN, the rest of the baseline methods are designed for cross network node classification problems.

- **GCN** [37]: A graph neural network that uses a message-passing technique to learn the feature embedding of the nodes. The architecture has been incorporated into OTGCN as well as other graph neural networks [52] [35] [31]. The GCN implementation was written using code from the torch geometric library [24] and involved two GCN layers

followed by a fully connected classification layer.

- **AdaGCN** [20]: A CNNC technique which uses a series of GCN layers to learn separate embeddings for the source and target networks. It then performs domain adaptation by using an adversarial discriminator to force the two embedding into a shared domain. The AdaGCN implementation used in this chapter was taken directly from the author's provide source code[1]. No significant changes were made to the author's implementation, which consists of three GCN layers and a single layer discriminator.
- **ASN** [75]: Another CNNC approach which consists of two separate GCN variational autoencoders (VAE) for the source and target, a shared GCN encoder which looks at both the source and target, and a adversarial discriminator. Here we used source code provide by the author[2] with a small modification to correct for NaNs produced by the VAE reconstruction loss. The VAE reconstruction loss equation implemented in ASN includes a large exponential term followed by an aggregation step which can lead to NaNs if there is a large dataset or large values in VAE output. We addressed this issue by restricting values in VAE-generated embedding to between -10 and 10. Each of the three encoders in ASN were implemented with two GCN layers and the two decoders had a single layer.
- **ACDNE** [56]: A CNNC approach which uses special feature extraction layers to learn separate embeddings for the features and structure of each network. It then concatenates the two embeddings into a single source embedding and a single target embedding before sending both to an adversarial discriminator. We followed the authors source code[3] with this method as well and made no significant changes. Each of the four feature extractors consisted of two layers.

For fair comparison we implemented OTGCN with two NFT layers and two GCN layers, similar to most of the baseline methods. We use the macro- and micro-F1 scores[26] as

[1]https://github.com/daiquanyu/AdaGCN_TKDE
[2]https://github.com/yuntaodu/ASN
[3]https://github.com/shenxiaocam/ACDNE

our evaluation metrics when comparing the performance of different classification methods. We perform 10-fold cross validation to select the hyperparameters of OTGCN as well as the reported baselines. For OTGCN, we tune the hyperparameters for λ and θ as well as standard hyperparameters such as learning rate. Possible hyperparameter values were [0.01, 0.03, 0.05] for λ and [5, 10, 15] for θ. Since the target graph is completely unlabeled and has potentially different distribution than the source graph, this poses a significant problem for hyperparameter tuning. To our knowledge there is no ideal solution to this problem. For this work, the source dataset was split into 10 folds and the target dataset was *excluded* from the hyperparameter tuning process. In each pass the fold selected for validation was removed from the remainder of the source dataset and treated as a distinct target dataset. We followed this process to choose the best hyperparameters for both our method and the reported baselines.

For OTGCN the first and second NFT layers produced hidden embeddings of 32 and 64 units respectively. Similarly, the two GCN embeddings were also 32 and 64. The selected hyperparameters were $\lambda = 0.01$ and $\theta = 10$.

Each of ACDNE's feature extrators is build with two layer models which contain 64 and 32 hidden dimensions. We chose hyperparameter candidate values by referencing the paper associated with the work. Our tuning script selected 1×10^{-5} for the weight of the pairwise constraint loss and 1×10^{-5} for the weight of the l2 regularization term.

ADAGCN's three GCN layers have 64, 32 and 16 hidden dimensions and the discriminator has a single 16 unit hidden layer. We chose hyperparameter candidate values by looking at existing values within their source code for other datasets. There were three model-specific hyperparmeters for controlling various portion of their loss functions. Our tuning script selected 5×10^{-5} for the weight of the l2 loss, 1.0 for the weight of the Wasserstein loss, and 5.0 for the weight of the penalty loss.

ASN's encoders were all two layers with 64 and 32 hidden units. Once again we chose hyperparameter candidate values by looking at existing information within the source code.

Our tuning script chose 0.0001 for the weight of the difference loss, 1.0 for the weight of the domain loss, and 0.5 for the weight of the reconstruction loss.

After hyperparameter tuning each method was trained with the chosen hyperparameters ten times with distinct random seeds. Each time the model was trained on the entire labeled source dataset and evaluated on the target dataset. The mean and standard deviation of the resulting micro and macro F1 results are recorded in Table 3.3.

Method	**Macro F1**	**Micro F1**
GCN	0.50265 +/- 0.03876	0.55821 +/- 0.02234
AdaGCN	0.27853 +/- 0.18966	0.37112 +/- 0.22752
ASN	0.38712 +/- 0.09156	0.52537 +/- 0.04418
ACDNE	0.94310 +/- 0.04065	0.94328 +/- 0.04049
OTGCN	**0.97907 +/- 0.00733**	**0.97910 +/- 0.00731**

Table 3.3: Performance results on target dataset.

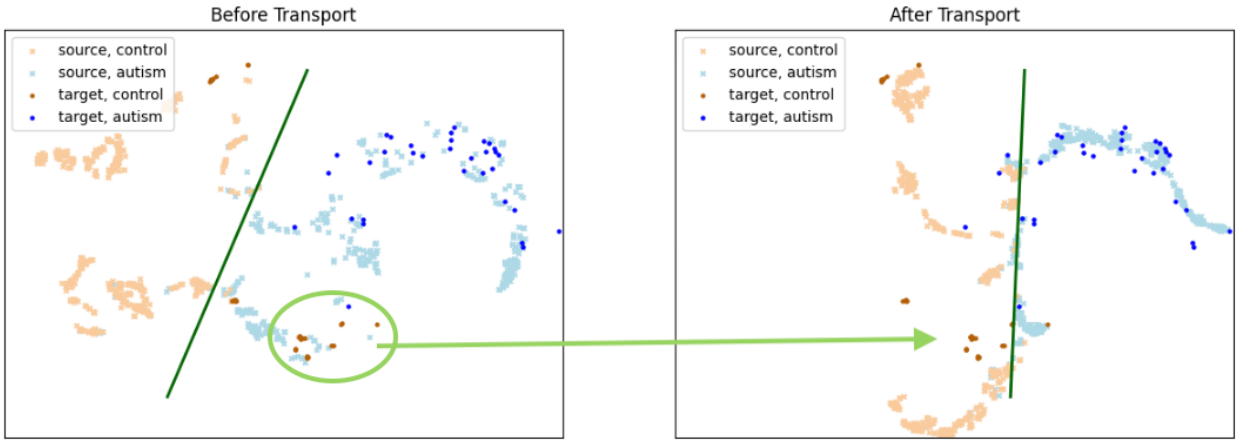


Figure 3.4: TSNE plot of combined embeddings before and after transport in the first pass of our OT layer. Points represent subjects and the lines are potential decision boundaries based off from each source dataset. The light colors represent the source dataset, before transport on the left and after transport on the right. The dark colors represent the target datset, which is the same in both cases. The green circle points out a portion of the target group which will likely be incorrectly classified before transportation occurs. The green arrow points to that same group on the right.

3.7.3 Results

Table 3.3 shows the micro- and macro-F1 results for OTGCN and all baselines on the prepared ABIDE target dataset. These results demonstrate the improved performance of OTGCN over existing CNNC baselines. The next best performing result was from ACDNE, which did not rely on GCN for it's graph embedding, but rather extracted structural and feature information separately into separate embeddings. OTGCN and ACDNE perfomed significantly better than all other baselines, likely because these two methods do not rely strictly upon GCN and the homophily principle.

Next we endeavor to establish the effectiveness of our optimal transport layer. To do this we save off a GCN embedding just before we send it to the very first OT transportation. We then transport the source embedding, use TSNE to reduce the embeddings to two dimensions and plot the results.

Fig. 3.4 is an example of a plot after this process. Here the points are colored according to label and dataset. On the left we see source before transportation in lighter colors, autism is light blue and control is light orange. The target dataset is plotted with it in darker colors, autism is dark blue and control is dark orange. A dark green line illustrates a potential decision boundary according to the source dataset, and it is clear that it will likely misclassify a significant group of the target samples. On the right we see a similar plot, but in this case the source samples have been transported to the target domain via our OT layer. Once again a dark green line illustrates a potential decision boundary based off from the source samples. It is clear from this demonstration that a decision boundary derived from the transported source embedding will more accurately classify subjects in the target network.

3.8 Conclusion

This chapter presents a deep learning framework called OTGCN to address the ASD detection problem using imaging and non-imaging information from multiple sites. OTGCN leverages ideas from graph neural networks to learn a feature representation of the nodes

and optimal transport to effectively tackle the domain adaptation problem between source and target graphs. Our framework also incorporates a nonlinear feature transformation layer to alleviate the issue of graphs with low homophily. We experimentally compared the performance of OTGCN against several state of the art CNNC baselines using the multi-site ABIDE dataset. These experiments demonstrated the superior performance of OTGCN over the baseline methods.

CHAPTER 4: FAIR CNNC VIA OPTIMAL TRANSPORT

4.1 Introduction

The previous chapter presented a method for leveraging optimal transport to address the concept drift challenge within the task of cross-network node classification (CNNC). In this chapter we extend the previous method to to address the challenges of fairness and scalability within CNNC.

Incorporating fairness consideration into node classification is a necessary challenge that must be addressed to prevent the model from generating biased prediction results. Graph neural networks (GNNs) are especially prone to fairness issues due to an artifact of the homophily principle, which states that similar nodes tend to be connected to each other [46]. GNNs rely upon this principle as they use message-passing to incorporate neighborhood information into the learning of node representations. Past research on graph fairness have, however, shown that neighborhood structure is often more dependent on the protected attributes than the classification labels [21][55]. As a consequence, GNNs have the potential to exacerbate unfairness as the learned node embeddings may capture more information about a node's protected group than its class label [34], a phenomenon that is also known as *link bias* [19]. To alleviate such bias, the goal of many fair node classification algorithms is to learn an embedding that is less reliant upon an individual's protected group [3][19][55]. However, despite the increasing research, current fairness-aware methods are mostly designed for homogeneous networks without accounting for the effect of concept drift.

Furthermore, despite their growing research, none of the existing CNNC approaches are designed to address fairness concerns. Similar to CNNC, while OT is a powerful approach for graph transfer learning [11], there has yet been any studies on incorporating fairness into the OT learning formulation. As OT maps samples in the source domain to similar samples in the target, it is also vulnerable to generating biased transportation plans. As previously noted, GNNs have an established weakness in that they often learn node representations which more closely represent protected groups than they do classification outcomes [55][34].

Thus, when used in conjunction with a GNN, OT has the potential to exacerbate unfairness by mapping between members of the same protected group instead of members of the same class.

One simple way to ensure fairness is by postprocessing the classifier trained on the feature embeddings generated by OT so they would not discriminate against particular subgroups of the population. However, this approach may not be as helpful since the post-processing does not alter the learned OT embedding that may still be biased. This begs the question: **How to effectively impart fairness into OT for fair cross-network node classification**?

In this chapter, we first present a novel framework called `FOCI` (**F**air Cr**O**ss-Network Node **C**lass**I**fication) that performs fair optimal transport while mitigating the concept drift issue in cross-network node classification. Specifically, `FOCI` considers the nodes' protected attribute information when transporting the source nodes to their corresponding target nodes when learning their feature embedding. We introduce a fair Sinkhorn distance measure for OT to encourage diversity in the mapping of the source nodes to target nodes. This strategy ensures that the learned features are oblivious to the protected attribute information, which is essential for creating an unbiased node feature representation.

After establishing the effectiveness of `FOCI` we develop and evaluate two additional variations. `FastFOCI` is the core `FOCI` approach, but with a few architectural improvements which allow it to scale to larger datasets. `SpFOCI` is a method which takes advantage of the performance improvements of `FastFOCI` to integrate statistical parity more directly into the optimal transport algorithm.

In summary, the main contributions of the paper include the following:

- We introduce a fair Sinkhorn distance measure to alleviate bias in the OT process by encouraging connections between members of distinct protected groups.
- We introduce a second fair Sinkhorn distance measure which aleviates bias by directly considering statistical parity within the optimal transport loss function.

- We developed `FOCI`, a graph neural network framework that uses the fair Sinkhorn distance measure for cross-network node classification. To the best of our knowledge, this is the first CNNC and OT method that considers fairness in its formulation.
- We develop `FastFOCI` and `SpFOCI`, two modifications which can be made on the core `FOCI` approach.
- We perform extensive experiments to demonstrate the effectiveness of `FOCI`, `FastFOCI`, and `SpFOCI` on several real world datasets.

4.2 Related Works

Previous chapters have disused foundational works in the areas of node classification, cross-network node classification and fairness in neural networks. This chapter brings these concepts together in fair cross-network node classification. To the best of our knowledge this is the first work addressing fair cross-network node classification and, therefore, there are no other related works in the space at this time. There is, however, a significant body of work addressing fairness in node classification tasks.

4.2.1 Fair Node Classification

As fairness has become an important factor to consider for node classification, various methods have recently been developed. For example, the FairDrop [58] method aims to reduce the negative effect of homophily by reducing the number of edges between nodes sharing the same sensitive attribute.Dai et al.[19] proposed FairGNN, which addresses the shortage of sensitive attributes for adversarial debiasing by estimating the users' sensitive attributes. Agarwal et al.[3] proposed the NIFTY algorithm, which enforces fairness and stability conditions on both the objective function and GNN architecture. However, none of these methods are designed for cross-network node classification, where there could be distribution shift among the networks.

4.3 Preliminaries

4.3.1 Problem Statement

Consider a set of graphs, $\mathcal{G}_1, \mathcal{G}_2, \cdots \mathcal{G}_n$, where each $\mathcal{G}_k(V_k, E_k, X_k, Y_k)$ is an attributed graph, with a node set V_k, edge set $E_k \subseteq V_k \times V_k$, node attribute matrix X_k, and node label vector Y_k. We further partition the columns of the node attribute matrix X_k into two submatrices, $X_k^{(p)}$ and $X_k^{(u)}$, where $X_k^{(p)}$ corresponds to the submatrix associated with the set of protected attributes (e.g., gender, race, age group, etc) while $X_k^{(u)}$ is the submatrix associated with the remaining (unprotected) attributes. Let $A^{(k)}$ denote the adjacency matrix representation of E_k, where $A_{ij}^{(k)} > 0$ if $(v_i, v_j) \in E_k$ and 0 otherwise.

In a domain adaptation setting, we assume there exists a partitioning on the set of the graphs into two disjoint groups—source graph $\mathcal{G}_s(V_s, E_s, X_s, Y_s)$, where the node labels in Y_s are known, and a target graph, $\mathcal{G}_t(V_t, E_t, X_t, Y_t)$, where the node labels in Y_t are unknown. Assuming the first m graphs contain labeled nodes, then $\mathcal{G}_s = \cup_{k=1}^{m} \mathcal{G}_k$ and $\mathcal{G}_t = \cup_{k=m+1}^{n} \mathcal{G}_k$, where $V_s = \cup_{k=1}^{m} V_k$, $E_s = \cup_{k=1}^{m} E_k$, $V_t = \cup_{k=m+11}^{n} V_k$, and $E_t = \cup_{k=m+1}^{n} E_k$. The adjacency matrices corresponding to the source and target graphs are denoted as $A_s \in \mathbb{R}^{n_s \times n_s}$ and $A_t \in \mathbb{R}^{n_t \times n_t}$, respectively, where $n_s = \sum_{i=1}^{m} |V_i|$ and $n_t = \sum_{i=m+1}^{n} |V_i|$ are the corresponding number of source and target nodes. Assuming the graphs have identical node features and the same set of class labels, therefore $X_s \in \mathbb{R}^{n_s \times d}$ and $X_t \in \mathbb{R}^{n_t \times d}$, $Y_s \in \{0, 1, \cdots, k-1\}^{n_s}$ and $Y_t \in \{0, 1, \cdots, k-1\}^{n_t}$, where k is the number of classes. At times, we also consider the combined graph $\mathcal{G}_c(V_c, E_c, X_c, Y_c)$, where $V_c = V_s \cup V_t$, $E_c = E_s \cup E_t$, $X_c = [X_s; X_t] \in \mathbb{R}^{n \times m}$, $A_c = [A_s \ \mathbf{0};\ \mathbf{0}^T \ A_t] \in \mathbb{R}^{n \times n}$, $Y_c \in \{0, 1, \cdots, k-1\}^n$, $n = n_s + n_t$, and $\mathbf{0}$ is an $n_s \times n_t$ null matrix.

Given a source graph, $\mathcal{G}_s = (V_s, E_s, X_s, Y_s)$ and target graph, $\mathcal{G}_t = (V_t, E_t, X_t, Y_t)$, the goal of fair cross-network node classification is to learn a target function, $f : V \rightarrow \{0, 1, \cdots, k-1\}$, that accurately classifies the labeled nodes in Y_s as well as the unlabeled nodes in Y_t while minimizing the disparity in the classification performance for different groups of the protected attributes.

4.3.2 Statistical Parity

We define fairness through the lens of statistical parity, which is one of the most commonly used fairness metrics. The metric states that the positive prediction outcomes should be evenly distributed between members of different protected groups. The metric is defined in the following equation where statistical parity is noted SP.

$$SP = |P(\hat{Y} = 1|X^{(p)} = 0) - P(\hat{Y} = 1|X^{(p)} = 1)|$$

where $\hat{Y}$ is a given prediction, $X^{(p)}$ is a protected attribute, and the goal is to have a value as close to zero as possible. Statistical parity is typically used in scenarios such as default prediction for a loan or insurance claims. In these scenarios specific protected groups are often disproportionately associated with a given label for complicated cultural reasons and improvements in statistical parity can be an important first step towards achieving equity.

4.4 Proposed FOCI Architecture

Fig. 4.1 shows a high-level overview of our proposed architecture. `FOCI` consists of 3 integrated modules for representation learning, OT for domain adaptation, and fully connected layers for node classification. Details of the architecture are given below.

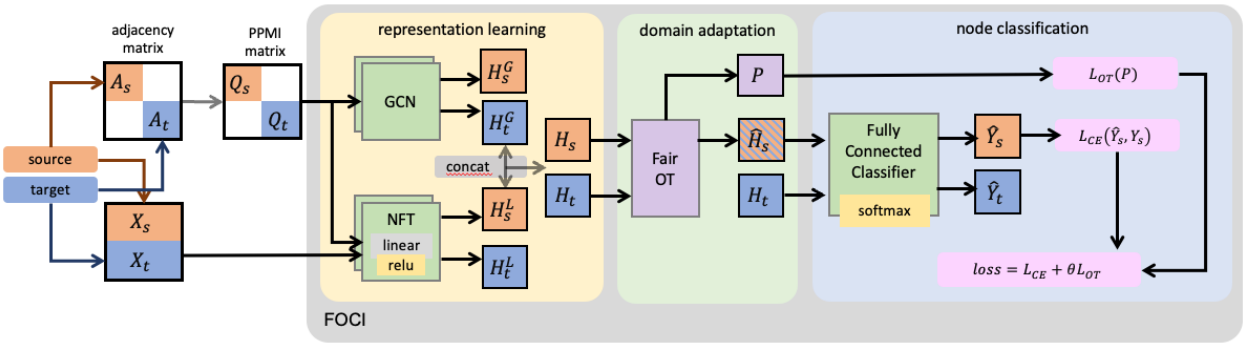


Figure 4.1: A schematic illustration of the FOCI framework. The framework contains integrated modules for representation learning (using 2-layer GCN with PPMI matrix along with 2-layer NFT), fair OT for domain adaptation, and a fully connected node classification layer.

4.4.1 Representation Learning & Pretraining

`FOCI` modifies the GCN architecture [37] to learn a joint embedding of the nodes in the source and target networks. The modifications are needed for two reasons. First, due to oversmoothing effect [40], current GCN cannot be easily extended beyond 2 or 3 layers, thus throttling the effective neighborhood size of its message passing. Because of its limited neighborhood size and the link bias problem noted in the introduction, the learned representation may inadvertantly encode the protected attribute information [19]. To overcome this limitation, we expand the neighborhood utilized by the GCN through the use of the *positive pointwise mutual information* (PPMI) matrix [39]. A PPMI matrix measures the topological proximity of nodes within some K-steps within the network and has been used with other GNN approaches [9, 57, 56, 75] to expand the neighborhood of interest. By using a PPMI matrix, the embedding learned will more likely capture information from nodes beyond those from the same protected group.

The representation learning module also learns a separate nonlinear embedding of the node features. This allows us to maintain a node embedding that is free of link bias for later use. We refer to this as the *node feature transformation* (NFT) layer, which consists of a fully-connected linear layer with ReLU activation function. We pre-train the network to classify only the labeled nodes in the source network. Specifically, `FOCI` uses two GCN layers with the PPMI matrix and two NFT layers. The learned node embeddings are concatenated as shown in Fig. 4.1 before being sent to a fully connected classification layer, which is trained to minimize the following cross-entropy loss:

$$\text{Pre-training:} \quad \mathcal{L}_{CE} = -\frac{1}{n_s} \sum_{x_i \in X_s} \sum_{j=0}^{l} Y_{ij} \log(\hat{Y}_{ij}) \tag{8}$$

4.4.2 Fair Optimal Transport

Our strategy to incorporate fairness into OT for graph transfer learning is by encouraging mappings between members of different protected groups. This heuristic helps to alleviate the link bias problem due to the inherent homophily effect in network data. This

method will be integrated directly into OT, resulting in a new fair Sinkhorn algorithm.

Let $X_s^{(p)}$ and $X_t^{(p)}$ be the protected attributes of the nodes in the source and target networks, respectively. We first create two sparse matrices, $R \in [0,1]^{n_s \times n_t}$ and $S \in [0,1]^{n_s \times n_t}$, where

$$R_{ij} = \begin{cases} 1 & \text{if } X_{s,i}^{(p)} = X_{t,j}^{(p)} \\ 0 & \text{otherwise} \end{cases}, \quad S_{ij} = \begin{cases} 1 & \text{if } X_{s,i}^{(p)} \neq X_{t,j}^{(p)} \\ 0 & \text{otherwise} \end{cases}$$

Given a transportation plan matrix P, we compute the following fairness loss:

$$\ell_F = \frac{\langle P, R \rangle_F}{\langle R, R \rangle_F} - \frac{\langle P, S \rangle_F}{\langle S, S \rangle_F},$$

where $\langle A, B \rangle_F = \sum_{ij} A_{ij} B_{ij}$ denotes the Frobenius inner product between two matrices. The smaller the fairness loss, the greater the emphasis is on transportation between samples in different protected groups. This enables us to incorporate fairness consideration into OT by introducing the following γ-fair Sinkhorn distance:

$$W_\gamma(\mu_s, \mu_t) \;=\; \langle P^\gamma, C \rangle \tag{9}$$

where

$$\begin{aligned} P^\gamma &= \text{argmin}_{P \in U(\mu_s, \mu_t)} \mathcal{L}_{OT}, \\ \mathcal{L}_{OT} &= \langle P, C \rangle - \lambda h(P) + \gamma \left[\frac{\langle P, R \rangle}{\langle R, R \rangle} - \frac{\langle P, S \rangle}{\langle S, S \rangle} \right] \\ U(\mu_s, \mu_t) &:= \left\{ P \in \mathbb{R}_+^{n_s \times n_t} \middle| P 1_{n_t} = \mu_s, P^T 1_{n_s} = \mu_t \right\} \end{aligned}$$

Armed with this new formulation we introduce the following theorem.

Theorem 1. *Given the γ-fair Sinkhorn distance in Eqn. 9, the solution for P^γ can be simplified as*

$$P^\gamma = diag(u) K \, diag(v)$$

where

$$K = e^{-\frac{1}{\lambda}(C + \gamma(\frac{R}{n_1} - \frac{S}{n_2}))}, u = e^{-\frac{1}{2} - \frac{1}{\lambda}\alpha}, v = e^{-\frac{1}{2} - \frac{1}{\lambda}\beta}$$

The proof for Theorem 1 is as follows.

Proof. The Lagrangian of the function in Eqn. 9 is

$$\begin{aligned}\mathcal{L} &= \sum_{ij}\left[P_{ij}C_{ij} + \lambda P_{ij}\log P_{ij} + \gamma P_{ij}\left(\frac{R_{ij}}{n_1} - \frac{S_{ij}}{n_2}\right)\right] \\ &\quad + \alpha^T(P1_d - \mu_s) + \beta^T(P^T 1_d - \mu_t)\end{aligned}$$

where $n_1 = \langle R, R\rangle$ and $n_2 = \langle S, S\rangle$. The solution can be found by taking the derivative of the Lagrangian function with respect to P_{ij} and setting it to zero.

$$P_{ij} = e^{-\frac{1}{2}-\frac{1}{\lambda}\alpha_i} e^{-\frac{1}{\lambda}(c_{ij}+\gamma(\frac{r_{ij}}{n_1}-\frac{s_{ij}}{n_2}))} e^{-\frac{1}{2}-\frac{1}{\lambda}\beta_j}$$

The theorem follows by replacing the terms for K, u, and v, respectively, into the above equation. □

Theorem 1 enables us to compute P^γ using the modified fair Sinkhorn algorithm shown in Algorithm 1, which in turn, allows us to find a fairness-aware transportation plan. These changes can be seen in Algorithm 1 and has no significant impact on computational complexity of existing Sinkhorn algorithm [18]. Additionally, since we are still utilizing squared l_2 loss for our cost and uniform marginals, the transported source node representation by OT remains unchanged as shown below:

$$\hat{H}^s = n_s P^\lambda H_t \tag{10}$$

4.4.3 Cross-Network Node Classification

The fair OT module will generate a transported source node embedding matrix, $\hat{H}^s$ using equation (10). Next $\hat{H}_s$ and the un-transformed target embedding H_t are passed to a fully connected network for node classification. After pre-training, the entire network is trained end-to-end to optimize the following joint objective function, which is a combination of the cross-entropy loss and optimal transport loss.

$$\mathcal{L} = \mathcal{L}_{CE} + \theta\mathcal{L}_{OT} \tag{11}$$

where θ is a hyperparameter that controls the tradeoff between the two losses.

Algorithm 1 Fair Sinkhorn

Require: $H_s, H_t, X_s^{(p)}, X_t^{(p)}, \lambda, \gamma$

$C = \text{computeCost}(H_s, H_t)$

$\mu_s, \mu_t = \text{computeUniformMarginals}(H_s, H_t)$

$R, S = \text{getMasks}(g_s, g_t)$

$K = e^{-\frac{1}{\lambda}(C+\gamma(\frac{R}{n_1}-\frac{S}{n_2}))}$ ▷ Modification

$u = \text{ones}(length(\mu_s))/length(\mu_s)$

$\tilde{K} = \text{diag}(1/\mu_s)K$

while u changes **do**

$u = 1/(\tilde{K}(\mu_t/(K'u))$

end while

$v = \mu_t/(K'u)$

$W = \text{sum}(u \odot ((K \odot C)v))$ ▷ Distance Measure

$P = \text{diag}(u)K\text{diag}(v)$ ▷ Transport Plan

$\hat{H}_s = n_s P H_t$

4.5 Proposed FOCI Modifications

4.5.1 `FastFOCI`

The first obvious hindrance to performance in `FOCI` is that the representation learning step must evaluate both the source and target graphs at the same time. GCN has a computational complexity of $O(|E|mh_1h_2)$ where $|E|$ is the number of edges, m is the number of

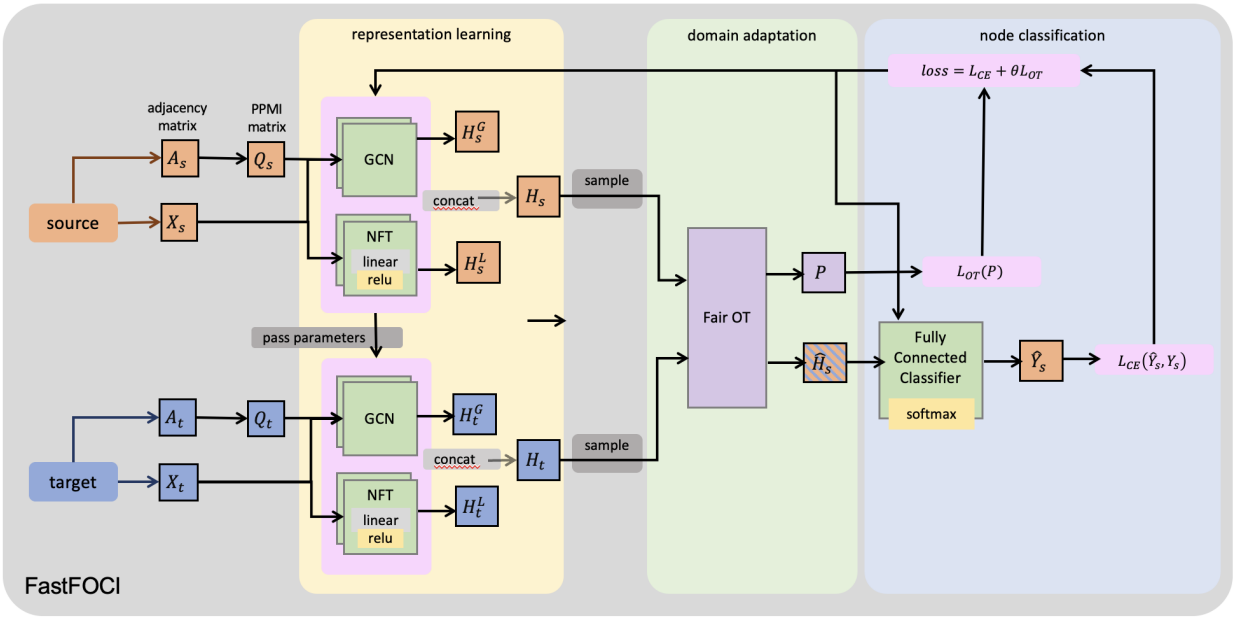


Figure 4.2: A schematic illustration of the `FastFOCI` framework.

Algorithm 2 `FOCI`

Require: $X_s, X_s^{(p)}, A_s, Y_s, X_t, X_t^{(p)}, A_t, \lambda, \gamma, \theta$
  $A_c, X_c = \text{Combine}(X_s, A_s, X_t, A_t)$
  $Q_c = \text{ComputePPMI}(A_c)$
  for `epoch in max_epochs` **do**
    $H_G = GCN(Q_c, X_c)$
    $H_L = NFT(X_c)$
    $H = H_G \bigoplus H_L$
    $H_s, H_t = \text{SeparateSourceAndTarget}(H)$
    if `epoch > pretrain` **then**
      $\hat{H}_s, P, R, S = \text{FairSink}(H_s, H_t, X_s^{(p)}, X_t^{(p)}, \lambda, \gamma)$
    end if
    $\hat{y}_s, \hat{y}_t = \text{Classifier}(\hat{H}_s, H_t)$
    if `epoch > pretrain` **then**
      $\text{ot\_loss} = \langle P, C \rangle - \lambda h(P) + \gamma \left(\frac{\langle P,R \rangle}{\langle R,R \rangle} - \frac{\langle P,S \rangle}{\langle S,S \rangle} \right)$
      $\text{loss} = \mathcal{L}_{\text{CE}}(y_s, \hat{y}_s) + \theta \mathcal{L}_{\text{OT}}$
    else
      $\text{loss} = \mathcal{L}_{\text{CE}}(y_s, \hat{y}_s)$
    end if
    update layer weights
  end for

input channels or features in X, h_1 is the number of hidden features in the first GCN layer, and h_2 is the number of features in the output layer [37]. While GCN is not horribly complex, it is linear in the number of graph edges, and therefore forcing a GCN to look at both graphs can have a significant impact. Therefore, the first major improvement implemented in `FastFOCI` is that the representation learning phase will consider the source data only.

Previously `FOCI` representation learning consisted of a GCN and an NFT which considered the source and target togeather. In `FastFOCI` the source graph will be fed to source GCN and NFT layers, GCN_s and NFT_s. In pretraining the output of these layers will be sent to a classifier and source predictions will be used to update the GCN_s, NFT_s, and classifier. In later steps GCN_s and NFT_s will first pass their current parameters to target GCN_t and NFT_t. Then GCN_s and NFT_s will be used to construct a source representation H_S while GCN_t and NFT_t is used to construct a target representation H_T. Later the loss will be used to update parameters of GCN_s, NFT_s, and classifier. In this way we reduce the number of

edges required by the GCN, which no longer requires full access to both source and target graphs at the same time.

The next area of improvement is in the optimal transport layer. The Sinkhorn algorithm has been experimentally shown to be $O(n_s n_t)$ [18]. The purpose of our optimal transport layer is to transport node representations from the source domain to the target domain prior to sending them to a classifier. The classifier does not need all node representations at the same time so we can improve the speed of the optimal transport layer by inserting a sampling step after node representations are acquired and before optimal transport occurs. Within the sampling step we randomly sample representations from both the source and target embeddings H_S and H_T described above and then pass those samples on to optimal transport. We can then transport the sampled source representations to send on to the classifier. These changes are all shown in Fig. 4.2.

4.5.2 `SpFOCI`

Previously the `FOCI` algorithm sought to improve fair treatment by intuitively encouraging transportation between members of different protected groups. Another option, however, is to incorporate statistical parity into the optimal transport loss function in a more direct manner. Remember that the formulation for statistical parity is the following:

$$P(\hat{Y} = 1 | X^{(p)} = 0) = P(\hat{Y} = 1 | X^{(p)} = 1)$$

According to our problem statement we have access to the source labels y_s as well as a binary protected attribute from the target dataset $X_t^{(p)} \in \mathbb{R}^{n_t}$ where $X_{ti}^{(p)}$ can be either 0 or 1. We now define $\tilde{y}_t = P^T y_s$, an approximation of $\hat{y}_t$ where $\tilde{y}_{ti}$ is the proportion of target sample i that is being mapped with source samples where $y_s = 1$. This approximation allows us to define a statistical parity measure below.

$$SP_{FOCI} = \left| \frac{(X_t^{(p)})^T \tilde{y}_t}{(X_t^{(p)})^T 1_{n_t}} - \frac{(1 - X_t^{(p)})^T \tilde{y}_t}{(1 - X_t^{(p)})^T 1_{n_t}} \right| = \left| \left(\frac{X_t^{(p)})^T}{(X_t^{(p)})^T 1_{n_t}} - \frac{(1 - X_t^{(p)})^T}{(1 - X_t^{(p)})^T 1_{n_t}} \right) P^T y_s \right|$$

Or for simplicity,

$$SP_{FOCI} = |w^T P^T y_s| \text{ where } w = \frac{X_t^{(p)}}{(X_t^{(p)})^T 1_{n_t}} - \frac{1 - X_t^{(p)}}{(1 - X_t^{(p)})^T 1_{n_t}}$$

We can now add this new term into the OT loss function and introduce a new SP-fair Sinkhorn distance:

$$W_{sp}(\mu_s, \mu_t) \quad = \quad \langle P^{sp}, C \rangle \tag{12}$$

where

$$\begin{aligned} P^{sp} &= \text{argmin}_{P \in U(\mu_s, \mu_t)} \mathcal{L}_{SP}, \\ \mathcal{L}_{SP} &= \langle P, C \rangle - \lambda h(P) + \gamma |w^T P^T y_s| \\ U(\mu_s, \mu_t) &:= \left\{ P \in \mathbb{R}_+^{n_s \times n_t} \middle| P 1_{n_t} = \mu_s, P^T 1_{n_s} = \mu_t \right\} \end{aligned}$$

This OT formulation leads us to the following theorem.

Theorem 2. *Given the SP-fair Sinkhorn distance in Eqn. 12, the solution for P^{SP} can be simplified as*

$$P^{SP} = diag(u) K \, diag(v) \text{ where } u = e^{-\frac{1}{2} - \frac{1}{\lambda}\alpha}, v = e^{-\frac{1}{2} - \frac{1}{\lambda}\beta} \text{ and}$$

$$K = \begin{cases} e^{-\frac{1}{\lambda}(C + \gamma w y_s^T)}, & \text{if } w^T P^T y_s > 0 \\ e^{-\frac{1}{\lambda}(C - \gamma w y_s^T)}, & \text{if } w^T P^T y_s < 0 \\ e^{-\frac{1}{\lambda}C}, & \text{if } w^T P^T y_s = 0 \end{cases} \tag{13}$$

The proof for Theorem 2 is as follows.

Proof. The Lagrangian of the function in Eqn. 12 is

$$\mathcal{L} \quad = \quad \sum_{ij} \lambda p_{ij} \log p_{ij} + p_{ij} c_{ij} + \gamma \left| \sum_{ij} w_i p_{ij} y_{sj} \right| + \alpha^T (P 1_d - \mu_s) + \beta^T (P^T 1_d - \mu_t)$$

dataset	nodes	edges	$\#(Y=1)$	$\#(X^{(p)}=1)$
Pokec-n	2933	16821.0	2145 (73%)	1722 (59%)
Pokec-z	3285	22454.0	2137 (65%)	1844 (56%)
Compas-0	2170	137473	1042 (48%)	1690 (78%)
Compas-1	1434	148012	674 (47%)	1140 (79%)
Abide-large	804	93886	370 (46%)	685 (85%)
Abide-small	67	587	33 (49%)	42 (63%)
Credit-0	10468	75669	8029 (76%)	5462 (52%)
Credit-1	10169	29227	8384 (82%)	5360 (52%)

Table 4.1: Breakdown of datasets used in experiments.

We can then solve the Lagrangian function with respect to P_{ij} to obtain the following result.

$$p_{ij} = \begin{cases} e^{-\frac{1}{2}-\frac{1}{\lambda}\alpha_i} e^{-\frac{1}{\lambda}(c_{ij}+\gamma w_i y_{sj})} e^{-\frac{1}{2}-\frac{1}{\lambda}\beta_j}, & \text{if } w^T P^T y_s > 0 \\ e^{-\frac{1}{2}-\frac{1}{\lambda}\alpha_i} e^{-\frac{1}{\lambda}(c_{ij}-\gamma w_i y_{sj})} e^{-\frac{1}{2}-\frac{1}{\lambda}\beta_j}, & \text{if } w^T P^T y_s < 0 \\ e^{-\frac{1}{2}-\frac{1}{\lambda}\alpha_i} e^{-\frac{1}{\lambda}c_{ij}} e^{-\frac{1}{2}-\frac{1}{\lambda}\beta_j}, & \text{if } w^T P^T y_s = 0 \end{cases} \tag{14}$$

The theorem follows by replacing the terms for K, u, and v, respectively, into the above equation. □

This new formulation allows us to obtain a solution by computing three different transport plans, one for each value of K, and then checking to see which one minimizes $\mathcal{L}_{SP}$. This approach will be more computationally expensive as we must run a Sinkhorn algorithm three times to obtain our three transport plans. Therefore, this OT approach is combined with the improvements made in `FastFOCI` to present `SpFOCI`.

4.6 Experimental Evaluation

This section describes the experiments performed to evaluate the effectiveness of our proposed `FOCI` framework. The source code for our approach can be found at https://github.com/ajoystephens/foci.

4.6.1 Data Preparation

We consider four real-world datasets for our experiments. **Pokec** [38] is a popular social media platform in Slovakia. The node feature information is obtained from the user profile

data while the link structure represents relationships between users. The classification task in this case is to predict whether a user smokes while the protected attribute here is user's sex. We use data from two separate geographical regions—Pokec-n and Pokec-z—to form the source and target networks for CNNC.

Compas [4] is a recidivism dataset in which each node corresponds to an incarcerated individual while an edge is formed by connecting individuals who were incarcerated during an overlapping time period. The classification task here is to predict whether an individual will re-offend again in the future. We use race as the protected attribute. The source and target networks are created by applying spectral clustering to split the network into 2 subgraphs, denoted as Compas-0 and Compas-1, respectively.

ABIDE [17] is a popular dataset for studying Autism Spectrum Disorder (ASD). Simular to previous works [51][52][35][31], we construct a population graph by using phenotypic subject data as node features and resting-state fMRI similarity to construct the edges. Here the presence of ASD is node label and sex is the protected attribute. The dataset was split into two separate graphs according to their data collection sites. Since many of the collection sites provide few or no female samples, we split the dataset into two by combining collection sites with over 70% male samples into a large source dataset and the remaining collection sites into a smaller target dataset. Given the small size of the ABIDE dataset and the need to have a large enough sample to train GNN models, there is only 1 version of the source and target networks in our experiments.

Credit [69] is a financial dataset where the task is to predict whether or not an individual will default on a loan. The protected attribute is the age of the individual, and edges are formed between individuals with similar spending and payment patterns. Credit-0 was selected from the initial 30,000 node dataset by first selecting the highest degree node and then repeatedly adding all immediate degree neighbors until the Credit-0 included over 10,000 nodes. Once Credit-0 was removed from the intial graph the process was repeated to obtain Credit-1.

4.6.2 Baseline Approaches

We compared `FOCI` against the following CNNC baseline methods.

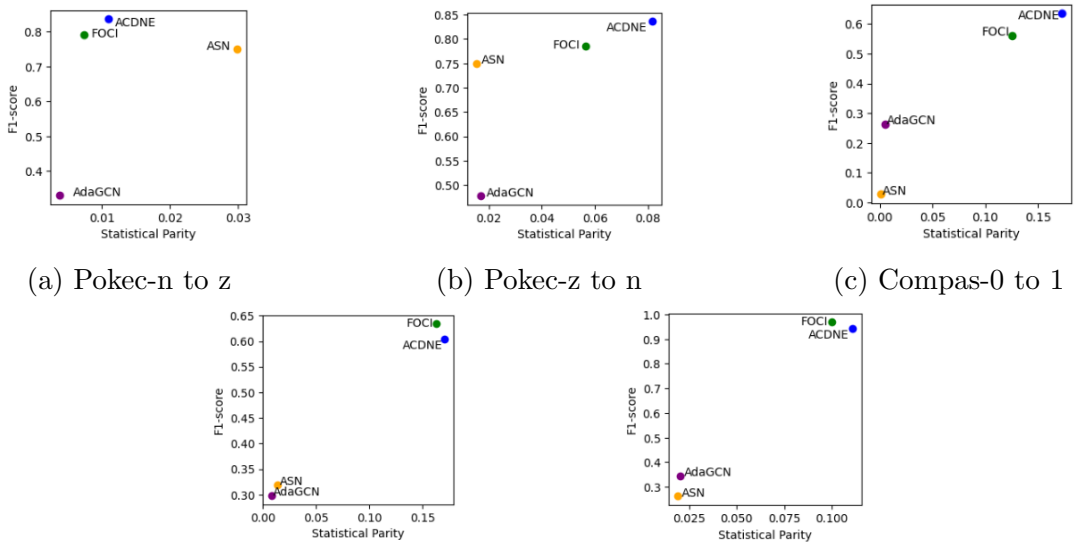
- **ACDNE** [56]: This approach uses an adversarial deep network embedding approach for domain adaptation. It learns a separate embedding from the node features and link structure before sending them to a discriminator. We use the implementation provided by the authors[4].
- **AdaGCN** [20]: This approach learns the node embeddings using a 3-layer GCN, followed by a single layer discriminator to address the domain drift issue. All experiments were performed using the source code made available by the authors[5].
- **ASN** [75]: This approach uses a series of 2-layer GCNs and GCN variational autoencoders (VAE) to learn the feature embedding. It addresses the domain adaptation issue using an adversarial discriminator. We use the authors' implementation[6] with a minor change using gradient clipping to address the NaN values generated by their VAEs.

For every method, we performed 10-fold cross fold validation on the source dataset to select their best hyperparameters. For the baseline methods, the range of their hyperparameter values include those reported in their authors' published papers and source code. Once the best hyperparameters were chosen, we acquired the final results by training the models on the full source dataset and applied them to the full target dataset. We repeated this 10 times with different random seeds and recorded the average F1-score and statistical parity values. Recall that an ideal F1-score should be 1 while statistical parity should be as close to 0 as possible.

4.6.3 Performance Comparison

The results comparing `FOCI` method to other CNNC approaches are shown in Fig. 4.3. Here the goal is to achieve high node classification results, as measured by F1-score, while

[4]https://github.com/shenxiaocam/ACDNE
[5]https://github.com/daiquanyu/AdaGCN_TKDE
[6]https://github.com/yuntaodu/ASN



(a) Pokec-n to z (b) Pokec-z to n (c) Compas-0 to 1

(d) Compas-1 to 0 (e) Abide

Figure 4.3: F1-score and statistical parity for FOCI and other CNNC baselines. The goal is to be closest to the upper left corner by balancing a high f1-score and low statistical parity.

simultaneously improving fairness, as measured by statistical parity. A good results moves closer to the upper left corner, achieving a high F1-score and low statistical parity. Both AdaGCN and ASN struggled to classify nodes in most of the datasets that we reviewed. They achieved very low F1-scores and low statistical parity by classifying the majority of nodes as the same class. We provide an example confusion matrix for one of the 10 runs of ASN in Table 4.2. Though the result suggests ASN has an excellent statistical parity, i.e., 0.0002, its F1-score is 0.1329, which is significantly lower than the average F1-score for `FOCI` (0.6637). This example illustrates how a model can achieve excellent statistical parity despite performing equally badly on all groups of the protected attribute. The exception to this is the Pokec datasets, where ASN manages to obtain reasonably high F1-score (though still lower than FOCI and ACDNE) and low statistical parity on Pokec-z to Pokec-n. Unfortunately, these good results were not consistent for ASN.

In each case, `FOCI` and ACDNE consistently achieve the two highest F1-scores, though the scores were relatively close. ACDNE had a slightly higher F1-score than `FOCI` in three of the five datasets. However, `FOCI` outperformed ACDNE in terms of statistical parity in all 5

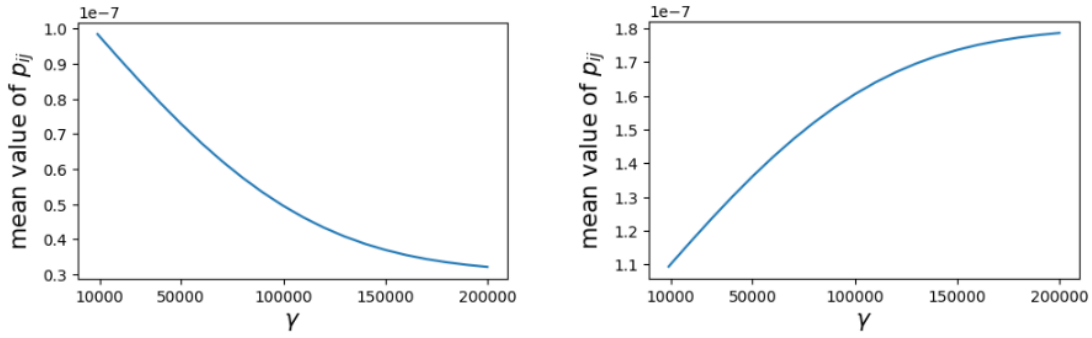
	$X^{(p)}=0$		$X^{(p)}=1$	
	$Y=0$	$Y=1$	$Y=0$	$Y=1$
$\hat{Y}=0$	422	298	438	569
$\hat{Y}=1$	32	26	40	47

Table 4.2: Example results from ASN on a target dataset of Compas-0 after training on Compas-1 over one seed. Though its statistical parity is 0.0002, its F1-score is poor (0.1329).

datasets. These results show that `FOCI` can perform comparably to other CNNC approaches in terms of their F1-scores while achieving better fairness.

4.6.4 Ablation Study

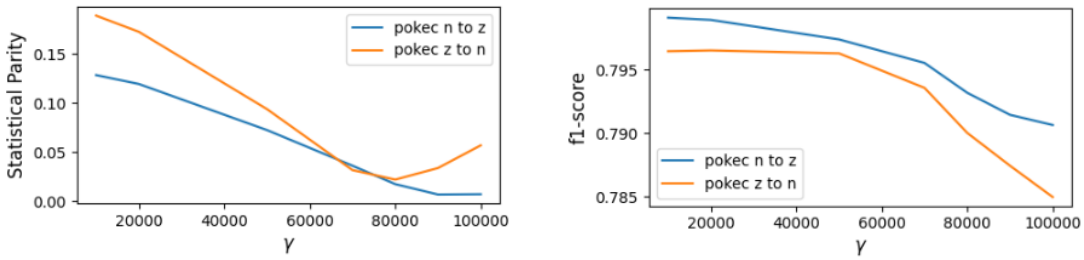
Here we investigate the impact of the introduced hyperparameter γ on the OT transport plan and model outcome. For this portion of the experiments we limit our exploration to the Pokec datasets and the FairSinkhorn OT algorithm as shown in Algorithm 1 and implemented in the `FOCI` method. In the first set of experiments we saved off hidden layers just before they were sent to OT and then performed OT on them with several different values of γ, evaluating the resulting transport plan matrix P^{γ}. In these and all other experiments involving OT we held $\lambda = 0.03$. Fig. 4.4 shows these results by plotting γ against the mean value of p_{ij} where nodes i and j do or do not share protected groups respectively. Note that



(a) same protected groups

(b) different protected groups

Figure 4.4: Impact of γ on transport plan matrix when mapping from pokec-n to pokec-z. As γ increases transport plan values decrease between nodes which share a protected group and increase for nodes which are in different protected groups.



(a) impact of γ on statistical parity

(b) impact of γ on f1-score

Figure 4.5: Impact of γ on `FOCI` model outcomes.

mean values for p_{ij} are very small as $P^{\gamma} \in \mathbb{R}^{m \times n}$.

Next we look at the impact of γ on model outcomes in terms of statistical parity and f1-score. For these experiments γ values were varied within the range that saw impact on P^{γ} in Fig. 4.4, but all other hyperparameters were kept the same. The models were trained with 10 random seeds for each value of γ and the resulting mean statistical parity and f1-scores were plotted in Fig 4.5.

Source	Target	Method	F1-Score	SP	Epoch Time (s)
pokec-n	pokec-z	`FOCI`	0.790 +/- 0.004	0.007 +/- 0.006	0.569 +/- 0.067
pokec-n	pokec-z	`FastFOCI`	0.773 +/- 0.041	0.013 +/- 0.014	0.117 +/- 0.042
pokec-n	pokec-z	`SpFOCI`	0.700 +/- 0.067	0.045 +/- 0.026	1.162 +/- 1.614
pokec-z	pokec-n	`FOCI`	0.784 +/- 0.008	0.056 +/- 0.016	0.627 +/- 0.065
pokec-z	pokec-n	`FastFOCI`	0.756 +/- 0.071	0.047 +/- 0.027	0.110 +/- 0.029
pokec-z	pokec-n	`SpFOCI`	0.672 +/- 0.072	0.068 +/- 0.054	2.481 +/- 1.672
compas-0	compas-1	`FOCI`	0.560 +/- 0.187	0.122 +/- 0.041	0.277 +/- 0.024
compas-0	compas-1	`FastFOCI`	0.599 +/- 0.085	0.158 +/- 0.027	0.108 +/- 0.022
compas-0	compas-1	`SpFOCI`	0.578 +/- 0.025	0.079 +/- 0.030	0.111 +/- 0.027
compas-1	compas-0	`FOCI`	0.638 +/- 0.009	0.163 +/- 0.011	0.177 +/- 0.021
compas-1	compas-0	`FastFOCI`	0.622 +/- 0.037	0.169 +/- 0.026	0.084 +/- 0.023
compas-1	compas-0	`SpFOCI`	0.627 +/- 0.018	0.181 +/- 0.017	0.086 +/- 0.023
credit-0	credit-1	`FastFOCI`	0.883 +/- 0.001	0.014 +/- 0.001	0.123 +/- 0.089
credit-0	credit-1	`SpFOCI`	0.910 +/- 0.001	0.026 +/- 0.011	0.194 +/- 0.162
credit-1	credit-0	`FastFOCI`	0.868 +/- 0.009	0.006 +/- 0.004	0.145 +/- 0.195
credit-1	credit-0	`SpFOCI`	0.866 +/- 0.008	0.020 +/- 0.018	0.263 +/- 0.527

Table 4.3: Comparison of `FOCI`, `FastFOCI` and `SpFOCI` in terms of F1-Score, statically parity (SP), and epoch time in seconds.

4.6.5 FOCI Comparison

Lastly we compare results from FOCI, FastFOCI and SpFOCI in terms of F1-score, statistical parity, and run time. The results for this comparison can be seen in Table 4.3. FOCI results for the credit datasets are excluded from this table because the FOCI method was unable to process the credit dataset. FOCI's optimal transport layer encountered numerical errors when attempting to transport the entire source and target credit datasets.

Here we see quickly see that FastFOCI provides significant run time improvements over FOCI, but with a slight decline in model utility and fairness. This suggests that FastFOCI may be the best option with larger datasets, but that FOCI may be the better choice with a smaller dataset. On the other hand, SpFOCI did not see an improvement in fairness over FOCI or FastFOCI in most cases.

4.7 Conclusion

This chapter presents a framework called FOCI for fair cross-network node classification. The framework uses a novel fair Sinkhorn distance measure to encourage mapping between members of different protected groups in the source and target networks. It then presents FastFOCI, a incremental improvement to FOCI which improves performance and scalablilty, and SpFOCI, a combination of FastFOCI and an alternative fair OT approach. We have experimentally shown that all three proposed approaches help to mitigate unfairness while maintaining accuracy comparable to other state-of-the-art CNNC baselines. We have also presented run time values comparing our three approaches and demonstrating the performance improvements of FastFOCI .

CHAPTER 5: CONCLUSION & FUTURE WORK

In this thesis I present several approaches for using optimal transport for domain adaptation in the CNNC task. OTGCN was introduced in Chapter 3 and provides a basic CNNC approach which utilizes optimal transport for domain adaptation and a GCN for node representation learning. I then expanded upon the OTGCN framework to develop `FOCI` , an intuitively fair CNNC framework in Chapter 4. `FastFOCI` and `SpFOCI` were then introduced as improvement addressing performance and offering an alternative fairness approach. OTGCN and `FOCI` were compared to other state-of-the-art CNNC baselines. `FOCI`, `FastFOCI` and `SpFOCI` were all compared on the largest of our datasets to explore scalabilty and performance.

Two immediate areas for further exploration are in the application of statistical parity and the sampling method. Perhaps `SpFOCI` could be improved by checking the transport plan within each iteration of the three Sinkhorn algorithms. Additionally, both `FastFOCI` and `SpFOCI` may see performance improvements with a more intelligent sampling approach such as active sampling [6] [25].

Another area to explore is methods for considering fairness directly within alternative domain adaptation approach. For example, instead of transporting source samples to the target domain it may be possible to generate an adapted training set using diffusion or other generative approaches

Lastly, a large area for future work is to design methods which target different definitions of fairness. In this thesis `FOCI` and `FastFOCI` implemented an intuitive fairness approach while `SpFOCI` addressed statistical parity, but, as discussed in Chapter 1, statistical parity is not always an ideal or applicable metric. Other metrics such as equalized odds or disparate impact are more applicable in some scenarios and may be explored in the CNNC setting.

REFERENCES

[1] T. Abrishami, N. Guillen, P. Rule, Z. Schutzman, J. Solomon, T. Weighill, and S. Wu. Geometry of graph partitions via optimal transport. *SIAM Journal on Scientific Computing*, 42(5):A3340–A3366, 2020.

[2] S. Abu-El-Haija, B. Perozzi, A. Kapoor, N. Alipourfard, K. Lerman, H. Harutyunyan, G. Ver Steeg, and A. Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *international conference on machine learning*, pages 21–29. PMLR, 2019.

[3] C. Agarwal, H. Lakkaraju, and M. Zitnik. Towards a unified framework for fair and stable graph representation learning. In *Uncertainty in Artificial Intelligence*, pages 2114–2124. PMLR, 2021.

[4] J. Angwin, J. Larson, S. Mattu, and L. Kirchner. Machine bias. propublica, may 23, 2016, 2016.

[5] Y. Bechavod and K. Ligett. Penalizing unfairness in binary classification. *arXiv preprint arXiv:1707.00044*, 2017.

[6] K. Bellare, S. Iyengar, A. G. Parameswaran, and V. Rastogi. Active sampling for entity matching. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1131–1139, 2012.

[7] R. Berk, H. Heidari, S. Jabbari, M. Joseph, M. Kearns, J. Morgenstern, S. Neel, and A. Roth. A convex framework for fair regression. *arXiv preprint arXiv:1706.02409*, 2017.

[8] S. Brody, U. Alon, and E. Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=F72ximsx7C1.

[9] S. Cao, W. Lu, and Q. Xu. Deep neural networks for learning graph representations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[10] R. Carette, F. Cilia, G. Dequen, J. Bosche, J.-L. Guerin, and L. Vandromme. Automatic autism spectrum disorder detection thanks to eye-tracking and neural network-based approach. In *Internet of Things (IoT) Technologies for HealthCare: 4th International Conference, HealthyIoT 2017, Angers, France, October 24-25, 2017, Proceedings 4*, pages 75–81. Springer, 2018.

[11] B. Chen, G. Bécigneul, O.-E. Ganea, R. Barzilay, and T. Jaakkola. Optimal transport graph neural networks. *arXiv preprint arXiv:2006.04804*, 2020.

[12] J. Chen, T. Ma, and C. Xiao. Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247*, 2018.

[13] L. Chen, Z. Gan, Y. Cheng, L. Li, L. Carin, and J. Liu. Graph optimal transport for cross-domain alignment. In *International Conference on Machine Learning*, pages 1542–1553. PMLR, 2020.

[14] B. Cheng, M. Liu, D. Shen, Z. Li, D. Zhang, and A. D. N. Initiative. Multi-domain transfer learning for early diagnosis of alzheimer's disease. *Neuroinformatics*, 15:115–132, 2017.

[15] S. Corbett-Davies, E. Pierson, A. Feller, S. Goel, and A. Huq. Algorithmic decision making and the cost of fairness. In *Proceedings of the 23rd acm sigkdd international conference on knowledge discovery and data mining*, pages 797–806, 2017.

[16] N. Courty, R. Flamary, and D. Tuia. Domain adaptation with regularized optimal transport. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part I 14*, pages 274–289. Springer, 2014.

[17] C. Craddock, S. Sikka, B. Cheung, R. Khanuja, S. S. Ghosh, C. Yan, Q. Li, D. Lurie, J. Vogelstein, R. Burns, et al. Towards automated analysis of connectomes: The configurable pipeline for the analysis of connectomes (c-pac). *Front Neuroinform*, 42:10–3389, 2013.

[18] M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper/2013/file/af21d0c97db2e27e13572cbf59eb343d-Paper.pdf.

[19] E. Dai and S. Wang. Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 680–688, 2021.

[20] Q. Dai, X. Shen, X.-M. Wu, and D. Wang. Network transfer learning via adversarial domain adaptation with graph convolution. *arXiv preprint arXiv:1909.01541*, 2019.

[21] Y. Dong, O. Lizardo, and N. V. Chawla. Do the young live in a "smaller world" than the old? age-specific degrees of separation in a large-scale mobile communication network. *arXiv preprint arXiv:1606.07556*, 2016.

[22] U. Erkan and D. N. Thanh. Autism spectrum disorder detection with machine learning methods. *Current Psychiatry Research and Reviews Formerly: Current Psychiatry Reviews*, 15(4):297–308, 2019.

[23] S. Ferradans, N. Papadakis, G. Peyré, and J.-F. Aujol. Regularized discrete optimal transport. *SIAM Journal on Imaging Sciences*, 7(3):1853–1882, 2014.

[24] M. Fey and J. E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[25] J. Gottlieb. Understanding active sampling strategies: Empirical approaches and implications for attention and decision research. *Cortex*, 102:150–160, 2018.

[26] M. Grandini, E. Bagli, and G. Visani. Metrics for multi-class classification: an overview. *arXiv preprint arXiv:2008.05756*, 2020.

[27] H. Guan, L. Wang, and M. Liu. Multi-source domain adaptation via optimal transport for brain dementia identification. In *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pages 1514–1517. IEEE, 2021.

[28] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[29] M. Hardt, E. Price, and N. Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

[30] M. HassanPour Zonoozi and V. Seydi. A survey on adversarial domain adaptation. *Neural Processing Letters*, 55(3):2429–2469, 2023.

[31] H. Jiang, P. Cao, M. Xu, J. Yang, and O. Zaiane. Hi-gcn: A hierarchical graph convolution network for graph embedding learning of brain network and brain disorders prediction. *Computers in Biology and Medicine*, 127:104096, 2020.

[32] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II 23*, pages 35–50. Springer, 2012.

[33] L. Kantorovitch. On the translocation of masses. *Management Science*, 5(1):1–4, 1958. doi: 10.1287/mnsc.5.1.1. URL https://doi.org/10.1287/mnsc.5.1.1.

[34] F. Karimi, M. Génois, C. Wagner, P. Singer, and M. Strohmaier. Homophily influences ranking of minorities in social networks. *Scientific reports*, 8(1):11077, 2018.

[35] A. Kazi, S. Shekarforoush, S. Arvind Krishna, H. Burwinkel, G. Vivar, K. Kortüm, S.-A. Ahmadi, S. Albarqouni, and N. Navab. Inceptiongcn: receptive field aware graph convolutional network for disease prediction. In *Information Processing in Medical Imaging: 26th International Conference, IPMI 2019, Hong Kong, China, June 2–7, 2019, Proceedings 26*, pages 73–85. Springer, 2019.

[36] S. Khoshraftar and A. An. A survey on graph representation learning methods. *arXiv preprint arXiv:2204.01855*, 2022.

[37] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[38] J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[39] O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems*, 27, 2014.

[40] Q. Li, Z. Han, and X.-M. Wu. Deeper insights into graph convolutional networks for semi-supervised learning, 2018.

[41] X. Li, Y. Gu, N. Dvornek, L. H. Staib, P. Ventola, and J. S. Duncan. Multi-site fmri analysis using privacy-preserving federated learning and domain adaptation: Abide results. *Medical Image Analysis*, 65:101765, 2020.

[42] X. Li, Y. Zhou, N. Dvornek, M. Zhang, S. Gao, J. Zhuang, D. Scheinost, L. H. Staib, P. Ventola, and J. S. Duncan. Braingnn: Interpretable brain graph neural network for fmri analysis. *Medical Image Analysis*, 74:102233, 2021.

[43] C. Lord, T. S. Brugha, T. Charman, J. Cusack, G. Dumas, T. Frazier, E. J. Jones, R. M. Jones, A. Pickles, M. W. State, et al. Autism spectrum disorder. *Nature reviews Disease primers*, 6(1):1–23, 2020.

[44] C. Louizos, K. Swersky, Y. Li, M. Welling, and R. Zemel. The variational fair autoencoder. *arXiv preprint arXiv:1511.00830*, 2015.

[45] B. T. Luong, S. Ruggieri, and F. Turini. k-nn as an implementation of situation testing for discrimination discovery and prevention. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 502–510, 2011.

[46] M. McPherson, L. Smith-Lovin, and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1):415–444, 2001.

[47] A. K. Menon and R. C. Williamson. The cost of fairness in binary classification. In *Conference on Fairness, accountability and transparency*, pages 107–118. PMLR, 2018.

[48] G. Monge. Mémoire sur la théorie des déblais et des remblais. *Mem. Math. Phys. Acad. Royale Sci.*, pages 666–704, 1781.

[49] H. S. Nogay and H. Adeli. Machine learning (ml) for the diagnosis of autism spectrum disorder (asd) using brain imaging. *Reviews in the Neurosciences*, 31(8):825–841, 2020.

[50] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[51] S. Parisot, S. I. Ktena, E. Ferrante, M. Lee, R. G. Moreno, B. Glocker, and D. Rueckert. Spectral graph convolutions for population-based disease prediction. In *Medical Image Computing and Computer Assisted Intervention- MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part III 20*, pages 177–185. Springer, 2017.

[52] S. Parisot, S. I. Ktena, E. Ferrante, M. Lee, R. Guerrero, B. Glocker, and D. Rueckert. Disease prediction using graph convolutional networks: application to autism spectrum disorder and alzheimer's disease. *Medical image analysis*, 48:117–130, 2018.

[53] D. Pessach and E. Shmueli. A review on fairness in machine learning. *ACM Computing Surveys (CSUR)*, 55(3):1–44, 2022.

[54] N. Quadrianto and V. Sharmanska. Recycling privileged learning and distribution matching for fairness. *Advances in neural information processing systems*, 30, 2017.

[55] T. Rahman, B. Surma, M. Backes, and Y. Zhang. Fairwalk: Towards fair graph embedding. 2019.

[56] X. Shen, Q. Dai, F.-l. Chung, W. Lu, and K.-S. Choi. Adversarial deep network embedding for cross-network node classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2991–2999, 2020.

[57] X. Shen, Q. Dai, S. Mao, F.-l. Chung, and K.-S. Choi. Network together: Node classification via cross-network deep network embedding. *IEEE Transactions on Neural Networks and Learning Systems*, 32(5):1935–1948, 2020.

[58] I. Spinelli, S. Scardapane, A. Hussain, and A. Uncini. Fairdrop: Biased edge dropout for enhancing fairness in graph representation learning. *IEEE Transactions on Artificial Intelligence*, 3(3):344–354, 2021.

[59] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu. A survey on deep transfer learning. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks*, pages 270–279. Springer, 2018.

[60] V. Titouan, N. Courty, R. Tavenard, and R. Flamary. Optimal transport for structured data with application on graphs. In *International Conference on Machine Learning*, pages 6275–6284. PMLR, 2019.

[61] T. Tong, K. Gray, Q. Gao, L. Chen, D. Rueckert, A. D. N. Initiative, et al. Multi-modal classification of alzheimer's disease using nonlinear graph fusion. *Pattern recognition*, 63:171–181, 2017.

[62] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *ICLR*, 2018.

[63] G. Wang, R. Ying, J. Huang, and J. Leskovec. Multi-hop attention graph neural network. *International Joint Conference on Artificial Intelligence*, 2021.

[64] J. Wang, L. Zhang, Q. Wang, L. Chen, J. Shi, X. Chen, Z. Li, and D. Shen. Multi-class asd classification based on functional connectivity and functional correlation tensor via multi-source domain adaptation and multi-view sparse representation. *IEEE transactions on medical imaging*, 39(10):3137–3147, 2020.

[65] Z. Wang, M. Ye, X. Zhu, L. Peng, L. Tian, and Y. Zhu. Metateacher: Coordinating multi-model domain adaptation for medical image classification. *Advances in Neural Information Processing Systems*, 35:20823–20837, 2022.

[66] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR, 2019.

[67] M. Wu, S. Pan, C. Zhou, X. Chang, and X. Zhu. Unsupervised domain adaptive graph convolutional networks. In *Proceedings of The Web Conference 2020*, pages 1457–1467, 2020.

[68] K. Xu, C. Li, Y. Tian, T. Sonobe, K.-i. Kawarabayashi, and S. Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pages 5453–5462. PMLR, 2018.

[69] I.-C. Yeh and C.-h. Lien. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert systems with applications*, 36(2):2473–2480, 2009.

[70] J. Yuan, C. Holtz, T. Smith, and J. Luo. Autism spectrum disorder detection from semi-structured and unstructured medical data. *EURASIP Journal on Bioinformatics and Systems Biology*, 2017:1–9, 2016.

[71] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. Learning fair representations. In *International conference on machine learning*, pages 325–333. PMLR, 2013.

[72] Y. Zeng, D. Cao, X. Wei, M. Liu, Z. Zhao, and Z. Qin. Multi-modal relational graph for cross-modal video moment retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2215–2224, 2021.

[73] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, 2021.

[74] J. Zhang, X. Xiao, L.-K. Huang, Y. Rong, and Y. Bian. Fine-tuning graph neural networks via graph topology induced optimal transport. *arXiv preprint arXiv:2203.10453*, 2022.

[75] X. Zhang, Y. Du, R. Xie, and C. Wang. Adversarial separation network for cross-network node classification. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2618–2626, 2021.

[76] J. Zhou, L. Liu, W. Wei, and J. Fan. Network representation learning: from preprocessing, feature extraction to node embedding. *ACM Computing Surveys (CSUR)*, 55 (2):1–35, 2022.

[77] A. Zunino, P. Morerio, A. Cavallo, C. Ansuini, J. Podda, F. Battaglia, E. Veneselli, C. Becchio, and V. Murino. Video gesture analysis for autism spectrum disorder detection. In *2018 24th international conference on pattern recognition (ICPR)*, pages 3421–3426. IEEE, 2018.