ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-6327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICASSP49357.2023.10095561

GRAPHMAD: GRAPH MIXUP FOR DATA AUGMENTATION USING DATA-DRIVEN CONVEX CLUSTERING

Madeline Navarro and Santiago Segarra

Electrical and Computer Engineering, Rice University, USA

ABSTRACT

We develop a novel *data-driven nonlinear mixup mechanism for graph data augmentation* and present *different mixup functions for sample pairs and their labels*. Mixup is a data augmentation method to create new training data by linearly interpolating between pairs of data samples and their labels. Mixup of graph data is challenging since the interpolation between graphs of potentially different sizes is an ill-posed operation. Hence, a promising approach for graph mixup is to first project the graphs onto a common latent feature space and then explore linear and nonlinear mixup strategies in this latent space. In this context, we propose to (i) project graphs onto the latent space of continuous random graph models known as graphons, (ii) leverage convex clustering in this latent space to generate nonlinear data-driven mixup functions, and (iii) investigate the use of different mixup functions for labels and data samples. We evaluate our graph data augmentation performance on benchmark datasets and demonstrate that nonlinear data-driven mixup functions can significantly improve graph classification.

***Index Terms*—** Graph mixup, convex clustering, graph classification, data augmentation, graph neural network.

1. INTRODUCTION

Rapid advancements in graph-based machine learning have led to the widespread use of graph neural networks (GNNs) in fields such as chemistry [1], wireless communications [2], and social network analysis [3]. However, as with existing deep learning models, GNNs require large datasets to prevent overfitting. Collecting more data is either costly or impossible, thus data augmentation arises as a cheap way to create more labeled samples from existing data. New data are generated to preserve label-dependent characteristics while presenting unseen views of data to help the model learn discriminative features.

Mixup poses an efficient method for augmenting Euclidean data such as images and text [4, 5]. In essence, mixup creates new data by selecting points along the linear interpolants between pairs of existing labeled samples. Formally, given two points $(\mathbf{x}_1, \mathbf{y}_1)$ and $(\mathbf{x}_2, \mathbf{y}_2)$ with data $\mathbf{x}_i$ and one-hot label vectors $\mathbf{y}_i$ for $i = 1, 2$, mixup generates new samples as

$$\mathbf{x}_{\text{new}} = \lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2, \tag{1a}$$

$$\mathbf{y}_{\text{new}} = \lambda \mathbf{y}_1 + (1-\lambda)\mathbf{y}_2, \tag{1b}$$

where the mixup parameter $\lambda \in [0, 1]$ selects new points along the connecting line. While mixup is theoretically and empirically effective at improving model generalization [6–8], it is not straightforward to translate conventional mixup to (non-Euclidean) graph data. Existing works [9–12] apply mixup in the graph domain by directly manipulating nodes, edges, or subgraphs; however, crucial structural information may be lost as classes can be sensitive to minor changes in graph topology. An attractive alternative is mixing graph data in a latent space, which includes mixup of learned graph representations [13, 14] or graph models [15].

Traditional mixup of Euclidean data, including graph mixup in latent space, applies the same linear function for data mixup (1a) and label mixup (1b) [16]. However, forcing linear mixing between classes may overlook crucial nonlinear behavior in the true distribution. Additionally, given a pair of samples, the user must decide where along the linear interpolant to sample new data. Previous works rely on empirically choosing the mixup parameter λ, but explicit investigation remains underexplored. Automatic data augmentation via learned optimal mixup parameters has attracted recent attention [7, 16, 17], but this incurs great computational cost.

We present *Graph Mixup for Augmenting Data (GraphMAD)* to augment graph data using data-driven mixup functions through convex clustering. We also analyze the choice of different mixup functions for data [cf. (1a)] and labels [cf. (1b)]. Our main contributions are as follows:

(i) We perform *nonlinear graph mixup in an interpretable continuous domain* given by *graphons*, random graph models that characterize graph structure.

(ii) We present convex clustering to *efficiently learn data-driven mixup functions*, where generated samples exploit relationships among all graphs as opposed to pairs of data.

(iii) We compare applying *different mixup functions for data samples and their labels* and demonstrate examples of datasets for which this is beneficial.

We introduce preliminaries on graph mixup and convex clustering in Section 2. Section 3 describes the three main steps of our proposed GraphMAD process for augmenting graph data. We demonstrate the superiority of our method in graph classification tasks on benchmark graph datasets in Section 4. Finally, our paper concludes on a discussion of the future directions in Section 5.

2. PRELIMINARIES AND RELATED WORK

In this section, we introduce three key topics related to our proposed GraphMAD method. We review mixup for graph data, graphons as continuous graph descriptors, and convex clustering.

2.1. Graph Mixup

Since its introduction [6], mixup has been popularly applied to image and text data [4, 5]. The application of mixup to graphs is a new but quickly developing area of research due to the ubiquity of graph data. Early research on graph mixup operates in the graph domain, directly

This work was supported by the NSF under award CCF-2008555.
Emails: nav@rice.edu, segarra@rice.edu

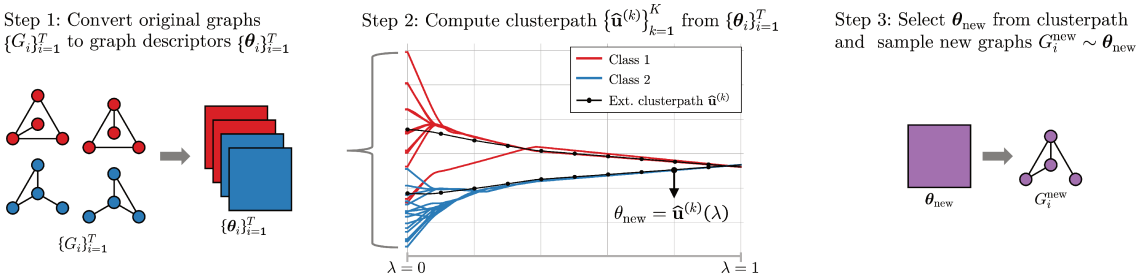

Fig. 1: Schematic of GraphMAD for graph data augmentation. **Step 1:** Convert each labeled graph $(G_i, \mathbf{y}_i)$ to a labeled graph descriptor $(\boldsymbol{\theta}_i, \mathbf{y}_i)$. **Step 2:** Compute the clusterpath $\{\hat{\mathbf{u}}^{(k)}\}_{k=1}^K$ and soft labels $\{\hat{\mathbf{y}}^{(k)}\}_{k=1}^K$ from the labeled descriptors $\{(\boldsymbol{\theta}_i, \mathbf{y}_i)\}_{i=1}^T$. **Step 3:** Sample $\boldsymbol{\theta}_{\text{new}}$ from a point in the clusterpath $\hat{\mathbf{u}}^{(k)}(\lambda)$ at $\lambda \in [0,1]$ and $k = 1, \ldots, K$ and generate new graphs G_i^{new} with label $\mathbf{y}_{\text{new}} = \hat{\mathbf{y}}^{(k)}(\lambda)$.

modifying the topology of the graphs [9, 10]. However, the cascading nature of graph connectivity means that crucial class-dependent structural information can be lost by changing a single edge or node. Thus, mixup for graphs in a latent graph representation space has gained popularity [13–15].

Authors in [15] present the closest work to our own. They estimate a set of K nonparametric random graph models known as *graphons* to collect discriminative characteristics for each of the K classes. Pairs of class graphons are then linearly interpolated to obtain new random graph models to create any number of new graph samples. However, because a single graphon represents all graphs in one class, the subsequent mixup cannot consider the spread of graphs within each class. Moreover, they consider only linear interpolation.

2.2. Continuous Graph Descriptors

To perform mixup of graphs in a continuous latent space as in [13–15], we require an Euclidean graph descriptor. In this work, similar to [15], we adopt the *graphon*, a bounded symmetric function $\mathcal{W} : [0,1]^2 \to [0,1]$ that can be interpreted as a random graph model associated with a family of graphs with similar structural characteristics [18–20]. To generate an undirected, unweighted graph $G = (\mathcal{V}, \mathcal{E})$ from a graphon $\mathcal{W}$, we require the following two steps

$$\zeta_i \sim \text{Uniform}([0,1]) \qquad \forall\, i \in \mathcal{V}, \tag{2a}$$

$$\mathbf{A}_{ij} = \mathbf{A}_{ji} \sim \text{Bernoulli}\left(\mathcal{W}(\zeta_i, \zeta_j)\right) \qquad \forall\, (i,j) \in \mathcal{V} \times \mathcal{V}, \tag{2b}$$

where $\mathbf{A} \in \{0,1\}^{|\mathcal{V}| \times |\mathcal{V}|}$ is the adjacency matrix with $\mathbf{A}_{ij} \neq 0$ if and only if $(i,j) \in \mathcal{E}$, and the latent variables $\zeta_i \in [0,1]$ are independently drawn for each node i. Intuitively, each node i is assigned a value ζ_i and the edge (i,j) is drawn between nodes i and j with probability $\mathcal{W}(\zeta_i, \zeta_j)$ for all $i \neq j$.

We convert every graph to a descriptor via graphon estimation, a well-studied task. A graphon can be inferred as a continuous function [21, 22] or as a coarser piecewise-constant stochastic block model (SBM) [23].

2.3. Convex Clustering

We apply convex clustering to characterize the spatial relationship of the graph data in graphon space. Convex clustering formulates the clustering problem as a continuous optimization problem, first introduced by [24–26] and inspired by sum-of-norms regularization [27]. Given a set of T data samples $\mathbf{x} = \{\mathbf{x}_i\}_{i=1}^T$ in $\mathbb{R}^p$, traditional convex clustering seeks to solve the regularized problem

$$\hat{\mathbf{u}}(\gamma) = \underset{\mathbf{u}}{\text{argmin}} \sum_{i=1}^{T} d_{\text{fid}}(\mathbf{u}_i, \mathbf{x}_i) + \gamma \sum_{i<j} w_{ij} d_{\text{fus}}(\mathbf{u}_i, \mathbf{u}_j), \tag{3}$$

where $\gamma \geq 0$ is the tunable fusion parameter, $w_{ij} \geq 0$ is the weight specifying the fusion strength between $\mathbf{u}_i$ and $\mathbf{u}_j$, and the functions d_{fid} and d_{fus} quantify data fidelity and sample fusion, respectively. We let the *clusterpath* be the solution path $\hat{\mathbf{u}} = \{\hat{\mathbf{u}}_i\}_{i=1}^T$ providing the cluster centroids $\hat{\mathbf{u}}(\gamma) = \{\hat{\mathbf{u}}_i(\gamma)\}_{i=1}^T$ for any $\gamma \geq 0$. If $\hat{\mathbf{u}}_i(\gamma) = \hat{\mathbf{u}}_j(\gamma)$ for $i \neq j$, then we say that samples $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same cluster at γ, where $\hat{K}(\gamma)$ is the total number of clusters for the fusion level γ. We show an example clusterpath in Step 2 of Fig. 1.

Under mild conditions, convex clustering enjoys theoretical guarantees for agglomerative fusions and cluster recovery due to the convexity of (3) [28–30]. However, convex clustering in non-Euclidean domains remains underexplored [31, 32]. Indeed, applying (3) for graph data suffers from the same difficulties as mixup of graphs, and imposing valid data formats may require nonconvex penalties, thus obviating the value of convex clustering.

3. METHODOLOGY

We let $G = (\mathcal{V}, \mathcal{E})$ denote an undirected, unweighted graph with set of nodes $\mathcal{V}$ and set of edges $\mathcal{E} = \mathcal{V} \times \mathcal{V}$. For a labeled graph G belonging to one of K classes, its label is represented by the one-hot vector $\mathbf{y} \in \{0,1\}^K$. When G belongs to the k-th class, $\mathbf{y}$ contains zeros everywhere except the k-th entry, which has the value 1. We denote the space of all graphs by $\mathcal{G}$.

3.1. GraphMAD

Given a set of T samples $\{(G_i, \mathbf{y}_i)\}_{i=1}^T$ of graph labeled data used to train a classifier $f : \mathcal{G} \to \{0,1\}^K$, our goal is to use GraphMAD to generate new data $\{(G_i^{\text{new}}, \mathbf{y}_i^{\text{new}})\}_{i=1}^{T'}$ so that a classifier f_{new} trained on the augmented dataset $\{(G_i, \mathbf{y}_i)\}_{i=1}^T \cup \{(G_i^{\text{new}}, \mathbf{y}_i^{\text{new}})\}_{i=1}^{T'}$ outperforms f when classifying unseen graphs. In particular, we focus on a GNN graph classifier [1, 33, 34], a highly effective model that requires a large training dataset.

We perform GraphMAD in graph descriptor space to obviate the difficulties of graph data for convex clustering and mixup described in Section 2. More specifically, we convert each graph G_i to an SBM graphon estimate [22] as introduced in Section 2.2. Given a set of graph descriptors denoted by $\boldsymbol{\theta} = \{\boldsymbol{\theta}_i\}_{i=1}^T$ and their respective one-hot labels $\mathbf{y} = \{\mathbf{y}_i\}_{i=1}^T$, we formulate the general mixup process of graph data as

$$\boldsymbol{\theta}_{\text{new}} = g_{\text{feat}}(\boldsymbol{\theta}; \lambda), \tag{4a}$$

$$\mathbf{y}_{\text{new}} = g_{\text{label}}(\boldsymbol{\theta}, \mathbf{y}; \lambda), \tag{4b}$$

where $\lambda \in [0,1]$ is the mixup parameter and g_{feat} and g_{label} are the mixup functions for graph descriptors and labels, respectively. For

Table 1: Graph classification accuracy on molecule and bioinformatics datasets. Columns g_{feat} and g_{label} are the data mixup for (4a) and the label mixup for (4b), respectively. The first row corresponds to the original datasets. The top performing methods are **bolded**.

Method		DD	PROTEINS	ENZYMES	AIDS	MUTAG	NCI109
Classes		2	2	6	2	2	2
Graphs		1178	1113	600	2000	188	4127
g_{feat}	g_{label}						
—	—	68.77 ± 2.35	$\mathbf{69.51 \pm 1.20}$	26.43 ± 2.55	96.18 ± 2.57	84.59 ± 5.53	68.23 ± 2.13
Linear (10)	Linear (1b) [15]	67.01 ± 1.72	65.15 ± 2.53	24.88 ± 3.38	96.82 ± 1.39	85.71 ± 7.15	68.16 ± 2.72
	Sigmoid (11a)	64.89 ± 1.49	68.42 ± 3.94	24.76 ± 4.10	96.07 ± 1.42	85.71 ± 4.63	65.96 ± 2.34
	Logit (11b)	66.22 ± 3.82	69.25 ± 2.94	25.95 ± 5.48	96.07 ± 1.27	80.08 ± 5.60	66.81 ± 4.07
	Clusterpath (8)	68.22 ± 3.71	69.38 ± 2.04	24.64 ± 2.39	95.86 ± 1.88	$\mathbf{87.22 \pm 4.96}$	65.01 ± 3.07
Clusterpath (7)	Linear (1b)	67.11 ± 1.56	67.51 ± 2.62	$\mathbf{26.67 \pm 6.49}$	$\mathbf{97.15 \pm 1.00}$	$\mathbf{87.24 \pm 4.21}$	$\mathbf{68.61 \pm 1.41}$
	Sigmoid (11a)	68.23 ± 3.61	64.60 ± 5.07	$\mathbf{32.62 \pm 6.35}$	97.07 ± 1.35	85.20 ± 3.53	67.50 ± 2.06
	Logit (11b)	$\mathbf{70.07 \pm 2.51}$	67.26 ± 2.84	25.71 ± 4.26	95.87 ± 1.47	80.10 ± 14.77	65.33 ± 3.35
	Clusterpath (8)	$\mathbf{70.44 \pm 3.79}$	$\mathbf{71.18 \pm 3.98}$	24.52 ± 3.30	$\mathbf{97.22 \pm 0.54}$	85.71 ± 5.40	$\mathbf{68.54 \pm 3.16}$

traditional mixup, we have that $g_{\text{feat}}(\mathbf{x}; \lambda) = \lambda \mathbf{x}_i + (1-\lambda)\mathbf{x}_j$ and $g_{\text{label}}(\mathbf{x}, \mathbf{y}; \lambda) = \lambda \mathbf{y}_i + (1-\lambda)\mathbf{y}_j$ for $i, j = 1, \ldots, T$, where $i \neq j$.

A schematic of our graph mixup method is presented in Fig. 1, which can be described in three steps: **Step 1:** We convert each graph G_i to a descriptor $\boldsymbol{\theta}_i = \boldsymbol{\theta}(G_i)$; **Step 2:** Given the descriptors $\boldsymbol{\theta}$, we compute the clusterpath $\hat{\mathbf{u}}$ over $\lambda \in [0, 1]$ through convex clustering; and **Step 3:** We choose $\lambda \in [0, 1]$ to select points in the clusterpath $\boldsymbol{\theta}_{\text{new}} = \hat{\mathbf{u}}(\lambda)$. The samples $\boldsymbol{\theta}_{\text{new}}$ in graph descriptor space are converted to the graph domain by sampling $G_i^{\text{new}} \sim \boldsymbol{\theta}_{\text{new}}$ from the graphon with label $\mathbf{y}_{\text{new}}$. The remainder of this section elaborates on each step of GraphMAD.

Step 1: Graph Descriptors. Convex clustering in Section 2.3 requires Euclidean data formats for the fidelity and fusion distances in (3). We convert the non-Euclidean graph data $\{G_i\}_{i=1}^T$ to descriptors from Section 2.2, denoted as $\boldsymbol{\theta}_i = \boldsymbol{\theta}(G_i)$ for samples $i = 1, \ldots, T$. Similarly to [15], our graph descriptors are SBM graphon approximations, where $\mathbf{W}_i \in [0, 1]^{D \times D}$ is obtained for each graph G_i by sorting and smoothing (SAS) [22] with D denoting the fineness of the graphon estimate. The graph descriptors are then set as $\boldsymbol{\theta}_i = \text{vec}(\mathbf{W}_i) \in [0, 1]^{D^2}$, where $\text{vec}(\mathbf{Y})$ denotes the vectorization of the matrix $\mathbf{Y}$. In this paper, we restrict our analysis to graphons, but GraphMAD accepts any descriptors that permit computation of (3) given choices of d_{fid} and d_{fus}.

Step 2: Data-Driven Mixup Function via Clusterpath. We approach graph mixup with the goal of exploiting the positions of data samples to inform where and how to interpolate between classes. We characterize the spatial relationships of graphs in descriptor space through the clusterpath, which we obtain by adapting (3) as

$$\hat{\mathbf{u}}(\lambda) = \underset{\mathbf{u}}{\text{argmin}} \sum_{i=1}^{T} \|\mathbf{u}_i - \boldsymbol{\theta}_i\|_2^2 + \frac{\lambda}{1-\lambda} \sum_{i<j} w_{ij} \|\mathbf{u}_i - \mathbf{u}_j\|_1, \tag{5}$$

where we specified the fidelity d_{fid} and fusion d_{fus} distances as the squared ℓ_2-norm and ℓ_1-norm, respectively, and we have $\lambda \in [0, 1]$ as in traditional mixup, where we apply (5) when $\lambda \in [0, 1)$ and we let $\lambda = 1$ correspond to minimizing the fidelity distance under the constraint that $\hat{\mathbf{u}}_i(1) = \hat{\mathbf{u}}_j(1)$ for every $i \neq j$. Section 2.3 describes the clusterpath $\hat{\mathbf{u}}$ as providing graphons $\hat{\mathbf{u}}(\lambda)$ for $\lambda \in [0, 1]$, where $\hat{\mathbf{u}}_i(0) = \boldsymbol{\theta}_i$ returns the original data and $\hat{\mathbf{u}}_i(1) = \frac{1}{T}\sum_{j=1}^T \boldsymbol{\theta}_j$ results in total fusion of the dataset for all $i = 1, \ldots, T$. Furthermore, we let the fusion weights $w_{ij} = 1$ when G_i and G_j belong to the same class, and $w_{ij} = \epsilon < 1$, otherwise. The labels inform the fusion weights to encourage tree-like clustering [30] while maintaining the data-dependent clusterpath shape. Note that the solutions to (5) lie in the convex hull of $\boldsymbol{\theta}$, resulting in valid symmetric and bounded graphons described in Section 2.2 and Step 1 of Section 3.1.

The clusterpath $\hat{\mathbf{u}} = \{\hat{\mathbf{u}}_i\}_{i=1}^T$ obtained via (5) results in T graphons that vary over $\lambda \in [0, 1]$. We compute a clusterpath of K branches by combining subsets of the T paths based on their cluster assignments from (5). Given a point λ^* where the number of clusters $\hat{K}(\lambda^*) = K$ as in Section 2.3, let the cluster assignments of $\hat{\mathbf{u}}(\lambda^*)$ be represented by index sets $\mathcal{I}^{(k)} \subseteq \{1, \ldots, T\}$ for $k = 1, \ldots, K$, where $\mathcal{I}^{(1)} \cup \cdots \cup \mathcal{I}^{(K)} = \{1, \ldots, T\}$. We collapse $\hat{\mathbf{u}}$ into K branches by averaging K graphon subsets based on $\{\mathcal{I}^{(k)}\}_{k=1}^K$ as

$$\hat{\mathbf{u}}^{(k)}(\lambda) = \frac{1}{|\mathcal{I}^{(k)}|} \sum_{i \in \mathcal{I}^{(k)}} \hat{\mathbf{u}}_i(\lambda) \quad \forall k = 1, \ldots, K, \ \lambda \in [0, 1]. \tag{6}$$

The *extended clusterpath* $\{\hat{\mathbf{u}}^{(k)}\}_{k=1}^K$ is superimposed on the original clusterpath in Step 2 of Fig. 1.

Step 3: Sampling. In this step, we describe the mixup processes in (4) using the extended clusterpath in (6) for sampling new data g_{feat} and new labels g_{label}.

New data $\boldsymbol{\theta}_{\text{new}}$. To obtain $\boldsymbol{\theta}_{\text{new}} = g_{\text{feat}}(\boldsymbol{\theta}; \lambda)$ from the extended clusterpath $\{\hat{\mathbf{u}}^{(k)}\}_{k=1}^K$ in (6), we first choose a branch $k \in \{1, \ldots, K\}$ and mixup parameter $\lambda \in [0, 1]$, and we obtain the graphon centroid as $\boldsymbol{\theta}_{new} = \hat{\mathbf{u}}^{(k)}(\lambda)$. A set of T' new graphs $\{G_i^{\text{new}}\}_{i=1}^{T'}$ is created from the graphon $\boldsymbol{\theta}_{new}$ as

$$G_i^{\text{new}} \sim \boldsymbol{\theta}_{\text{new}} = \hat{\mathbf{u}}^{(k)}(\lambda) \tag{7}$$

where $G_i^{\text{new}} \sim \boldsymbol{\theta}_{\text{new}}$ refers to the process of sampling a graph from a graphon as described in (2), so we can generate any number of new graphs for every k and λ that we consider.

New labels $\mathbf{y}_{\text{new}}$. We can obtain labels $\mathbf{y}_{\text{new}} = g_{\text{label}}(\boldsymbol{\theta}, \mathbf{y}; \lambda)$ using the extended clusterpath $\{\hat{\mathbf{u}}^{(k)}\}_{k=1}^K$ as with $\boldsymbol{\theta}_{\text{new}}$. To do this, we compute soft labels $\mathbf{y}_{\text{new}} \in [0, 1]^K$ using class proportions of samples $\boldsymbol{\theta}$ in each branch of the clusterpath $\hat{\mathbf{u}}^{(k)}$, as described below.

For each branch, we define a label clusterpath function $\hat{\mathbf{y}}^{(k)}(\lambda) = g_{\text{label}}(\boldsymbol{\theta}, \mathbf{y}; \lambda)$ that yields the soft label $\mathbf{y}_{\text{new}} = \hat{\mathbf{y}}^{(k)}(\lambda) \in [0, 1]^K$ corresponding to the graphon $\hat{\mathbf{u}}^{(k)}(\lambda)$. For the k-th branch, we let the l-th entry of $\hat{\mathbf{y}}^{(k)}(0) \in [0, 1]^K$ be the proportion of class l assigned to branch k, and we let $\hat{\mathbf{y}}^{(k)}(1) = \frac{1}{K}\mathbf{1}$, where $\mathbf{1}$ is the all-ones vector of length K, denoting total dataset fusion for equally-sized classes. We then compute $\hat{\mathbf{y}}^{(k)}(\lambda)$ for $\lambda \in [0, 1]$ as

$$\hat{\mathbf{y}}^{(k)}(\lambda) = g_{\text{cp}}^{(k)}(\lambda) \hat{\mathbf{y}}^{(k)}(0) + \left(1 - g_{\text{cp}}^{(k)}(\lambda)\right) \hat{\mathbf{y}}^{(k)}(1), \tag{8}$$

where $g_{\text{cp}}^{(k)}$ represents the rate of change of the k-th branch of the clusterpath $\hat{\mathbf{u}}^{(k)}$ with respect to the mixup parameter λ. More specifically, we have $g_{\text{cp}}^{(k)} : [0, 1] \to [0, 1]$ as

$$g_{\text{cp}}^{(k)}(\lambda) = \frac{\|\hat{\mathbf{u}}^{(k)}(0) + \int_0^\lambda \nabla_\lambda \hat{\mathbf{u}}^{(k)}(\tau) d\tau\|_2^2 - \|\hat{\mathbf{u}}^{(k)}(0)\|_2^2}{\|\hat{\mathbf{u}}^{(k)}(1)\|_2^2 - \|\hat{\mathbf{u}}^{(k)}(0)\|_2^2}, \tag{9}$$

Table 2: Counterpart of Table 1 for graph classification accuracy on social datasets.

Method		COLLAB	IMDB-B	IMDB-M
Classes		3	2	3
Graphs		5000	1000	1500
g_{feat}	g_{label}			
—	—	$\mathbf{80.00 \pm 0.96}$	73.14 ± 3.15	47.71 ± 4.25
Ln. (10)	Ln. (1b) [15]	77.60 ± 1.53	72.07 ± 2.06	47.24 ± 4.21
	Sig. (11a)	78.21 ± 1.16	$\mathbf{74.00 \pm 2.14}$	$\mathbf{49.67 \pm 2.15}$
	Log. (11b)	78.19 ± 1.61	72.64 ± 1.73	47.43 ± 2.45
	Cp. (8)	78.41 ± 0.99	71.43 ± 3.25	47.29 ± 5.21
Cp. (7)	Ln. (1b)	78.93 ± 2.63	70.57 ± 4.89	45.52 ± 4.09
	Sig. (11a)	77.89 ± 1.30	$\mathbf{75.00 \pm 5.13}$	44.48 ± 2.78
	Log. (11b)	$\mathbf{80.39 \pm 1.20}$	73.43 ± 4.75	48.76 ± 2.43
	Cp. (8)	79.55 ± 2.29	71.43 ± 4.72	$\mathbf{49.71 \pm 4.33}$

for $\lambda \in [0,1]$ and $k = 1, \ldots, K$, where $g_{\text{cp}}^{(k)}(\lambda) = \lambda$ for $\lambda \in \{0,1\}$.

Similarly to $\boldsymbol{\theta}_{\text{new}} = \hat{\mathbf{u}}^{(k)}(\lambda)$, new graph labels $\mathbf{y}_{\text{new}} = \hat{\mathbf{y}}^{(k)}(\lambda)$ for $\lambda \in [0,1]$ require selecting a branch $k \in \{1, \ldots, K\}$. When the clusterpath is used for both g_{feat} and g_{label}, that is, $\boldsymbol{\theta}_{\text{new}} = \hat{\mathbf{u}}^{(k)}(\lambda)$ in (7) and $\mathbf{y}_{\text{new}} = \hat{\mathbf{y}}^{(k)}(\lambda)$ in (8), we use the same branch k of the extended clusterpath $\hat{\mathbf{u}}^{(k)}$ for (7) and (9).

Under Steps 1 to 3, we obtain data-driven mixup functions for (4) using descriptors $\boldsymbol{\theta}$, where new samples $\boldsymbol{\theta}_{\text{new}} = g_{\text{feat}}(\boldsymbol{\theta}; \lambda)$ and labels $\mathbf{y}_{\text{new}} = g_{\text{label}}(\boldsymbol{\theta}, \mathbf{y}; \lambda)$ depend on the clusterpath in (6) for $\lambda \in [0,1]$. We sample graphs $\{G_i^{\text{new}}\}_{i=1}^{T'}$ from $\boldsymbol{\theta}_{\text{new}}$ via (7).

To conclude our GraphMAD discussion, we note that while both data mixup (7) and label mixup (8) depend on the clusterpath $\{\hat{\mathbf{u}}^{(k)}\}_{k=1}^{K}$ in (6), these mixup functions need not be applied together. For example, for some $\lambda \in [0,1]$ and $k = 1, \ldots, K$, we may have $\boldsymbol{\theta}_{\text{new}} = \hat{\mathbf{u}}^{(k)}(\lambda)$ from (7) and $\mathbf{y}_{\text{new}} = \lambda \hat{\mathbf{y}}^{(k)}(0) + (1-\lambda)\hat{\mathbf{y}}^{(k)}(1)$ as in (1b). In the next section we demonstrate GraphMAD data mixup via (7) and label mixup via (8), both implemented jointly with each other and with other choices of g_{feat} and g_{label}.

4. RESULTS

We demonstrate GraphMAD for creating labeled graph data to improve graph classification performance. As a graph classifier, we use the same GNN for all our experiments, the Graph Isomorphism Network (GIN) [34], which uses node spatial relations to aggregate neighborhood features. We compare classification performance for the original dataset with that of augmented datasets via multiple methods of graph mixup.

For mixup of graph data g_{feat}, we compare GraphMAD's clusterpath data mixup (7) with linear graphon mixup [15]. In this latter approach, we estimate one SBM graphon $\mathbf{W}^{(k)} \in [0,1]^{D \times D}$ for each class $k = 1, \ldots, K$. Then, for random pairs of classes $k, k' = 1, \ldots, K$ with $k \neq k'$, we linearly interpolate with $\lambda \in [0,1]$ and sample new graphs as

$$G_i^{\text{new}} \sim \mathbf{W}_{\text{new}} = \lambda \mathbf{W}^{(k)} + (1-\lambda)\mathbf{W}^{(k')}. \tag{10}$$

We consider additional variants for the label mixup g_{label}, apart from the classical linear function in (1b) and the proposed one in (8). In particular, given $\lambda \in [0,1]$ and label vectors $\mathbf{y}_i, \mathbf{y}_j \in [0,1]^K$, we also consider the mixup functions

$$\mathbf{y}_{\text{new}} = \text{sig}(\lambda)\mathbf{y}_i + (1 - \text{sig}(\lambda))\mathbf{y}_j, \tag{11a}$$

$$\mathbf{y}_{\text{new}} = \text{logit}(\lambda)\mathbf{y}_i + (1 - \text{logit}(\lambda))\mathbf{y}_j, \tag{11b}$$

where sigmoid refers to $\text{sig}(x) = 1/\big(1 + \exp\{-a(2x-1)\}\big)$ and logit is $\text{logit}(x) = \log\big(x/(1-x)\big)/2a + 1/2$ for $a > 0$ and $x \in [0,1]$. Note that $\mathcal{G}$-Mixup [15] has g_{feat} as (10) and g_{label} as (1b)

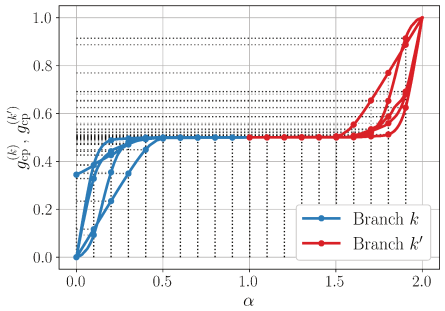

Fig. 2: Extended clusterpath behavior between branches k and k' for ENZYMES dataset. For $\alpha \in [0,2]$, we show the level of data mixup from branch k to fusion in blue as $\frac{1}{2} g_{\text{cp}}^{(k)}(\alpha)$ for $\alpha \in [0,1]$ and from fusion to branch k' in red as $1 - \frac{1}{2} g_{\text{cp}}^{(k')}(2-\alpha)$ for $\alpha \in [1,2]$.

We perform graph classification on nine graph benchmark datasets from the TUDatasets collection [35]: DD, PROTEINS, and ENZYMES for protein categorization, AIDS, MUTAG, and NCI109 for molecule classification, and COLLAB, IMDB-B, and IMDB-M for social network classification. In Tables 1 and 2 we present classification accuracy for each dataset using all pairs of data and label mixup functions, where g_{feat} is (7) or (10), and g_{label} is selected from (1b), (8), or (11). The first row with no choice of g_{feat} and g_{label} denotes performance using the original dataset. The mixup parameter $\lambda \sim \text{Unif}([0,1])$ is the same for all choices of g_{feat} and g_{label}.[1]

GraphMAD data mixup in (7) achieves the best performance for all nine datasets, and (7) is the only data augmentation method to improve performance for PROTEINS and COLLAB. Furthermore, for all *multi-class* datasets, since GraphMAD exploits relationships among all classes, it is always able to improve accuracy above the baseline compared to linear graphon mixup in (10), which only performs mixup between pairs of classes and thus is outperformed by the vanilla GIN for COLLAB and ENZYMES.

We examine the 6-class ENZYMES dataset, whose performance using GraphMAD data mixup in (7) and sigmoid label mixup in (11a) achieves an accuracy gap of around 6% over other methods. For topology-sensitive protein datasets such as ENZYMES, new data close in graphon space to the original data may produce incorrect graph structures for protein categorization, hence we wish to sample farther away from the original classes. This is automatically captured by the shape of the clusterpath. The data clusterpath across two branches $\hat{\mathbf{u}}^{(k)}$ and $\hat{\mathbf{u}}^{(k')}$ is shown in Fig. 2 for several choices of k and k'. For uniformly selected values of λ, new data $\boldsymbol{\theta}_{\text{new}}$ are close to the mean of the dataset for several values of λ. Thus samples generated through the data-driven GraphMAD are generated far from the original data, which dramatically improves performance.

5. CONCLUSION

We proposed a data-driven mixup function for graph data augmentation via convex clustering, where GraphMAD outperformed traditional linear mixup for several graph classification datasets. Potential directions for development include learning parameters for convex clustering, such as the fusion weights and the selected clusterpath branch for sampling. Furthermore, we can convert GraphMAD to a completely data-driven approach by using similarity-based fusion weights and applying self-supervised learning approaches.

[1]Further implementation details can be found in our code, provided at https://github.com/mn51/graphmad.

6. REFERENCES

[1] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Intl. Conf. on Machine Learning (ICML)*, pp. 1263–1272, PMLR, 2017.

[2] A. Chowdhury, G. Verma, C. Rao, A. Swami, and S. Segarra, "Unfolding WMMSE using graph neural networks for efficient power allocation," *IEEE Trans. Wireless Commun.*, vol. 20, no. 9, pp. 6004–6017, 2021.

[3] J. Qiu, J. Tang, H. Ma, Y. Dong, K. Wang, and J. Tang, "DeepInf: Social influence prediction with deep learning," in *Intl. Conf. on Knowledge Discovery and Data Mining (SIGKDD)*, pp. 2110–2119, 2018.

[4] D. Lewy and J. Mańdziuk, "An overview of mixing augmentation methods and augmentation strategies," *Artif. Intell. Review*, pp. 1–59, 2022.

[5] H. Guo, Y. Mao, and R. Zhang, "Augmenting data with mixup for sentence classification: An empirical study," *arXiv preprint arXiv:1905.08941*, 2019.

[6] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[7] H. Guo, Y. Mao, and R. Zhang, "Mixup as locally linear out-of-manifold regularization," in *AAAI Conf. on Artif. Intell.*, vol. 33, pp. 3714–3722, 2019.

[8] L. Zhang, Z. Deng, K. Kawaguchi, A. Ghorbani, and J. Zou, "How does mixup help with robustness and generalization?," in *Intl. Conf. on Learning Representations (ICLR)*, 2021.

[9] H. Guo and Y. Mao, "ifMixup: Towards intrusion-free graph mixup for graph classification," *arXiv preprint arXiv:2110.09344*, 2021.

[10] J. Park, H. Shim, and E. Yang, "Graph Transplant: Node saliency-guided graph mixup with local structure preservation," *arXiv preprint arXiv:2111.05639*, 2021.

[11] K. Ding, Z. Xu, H. Tong, and H. Liu, "Data augmentation for deep graph learning: A survey," *arXiv preprint arXiv:2202.08235*, 2022.

[12] T. Zhao, G. Liu, S. Günnemann, and M. Jiang, "Graph data augmentation for graph machine learning: A survey," *arXiv preprint arXiv:2202.08871*, 2022.

[13] Y. Wang, W. Wang, Y. Liang, Y. Cai, and B. Hooi, "Mixup for node and graph classification," in *Proc. World Wide Web Conf.*, pp. 3663–3674, 2021.

[14] V. Verma, M. Qu, K. Kawaguchi, A. Lamb, Y. Bengio, J. Kannala, and J. Tang, "GraphMix: Improved training of GNNs for semi-supervised learning," in *AAAI Conf. on Artif. Intell.*, vol. 35, pp. 10024–10032, 2021.

[15] X. Han, Z. Jiang, N. Liu, and X. Hu, "G-Mixup: Graph data augmentation for graph classification," *arXiv preprint arXiv:2202.07179*, 2022.

[16] H. Guo, "Nonlinear mixup: Out-of-manifold data augmentation for text classification," in *AAAI Conf. on Artif. Intell.*, vol. 34, pp. 4044–4051, 2020.

[17] Z. Mai, G. Hu, D. Chen, F. Shen, and H. T. Shen, "MetaMixUp: Learning adaptive interpolation policy of mixup with metalearning," *IEEE Trans. Neural Netw. and Learning Sys.*, 2021.

[18] L. Lovász, *Large networks and graph limits*, vol. 60. American Mathematical Soc., 2012.

[19] P. Diaconis and S. Janson, "Graph limits and exchangeable random graphs," *arXiv preprint arXiv:0712.2749*, 2007.

[20] M. Avella-Medina, F. Parise, M. T. Schaub, and S. Segarra, "Centrality measures for graphons: Accounting for uncertainty in networks," *IEEE Trans. Network Science and Eng.*, vol. 7, no. 1, pp. 520–537, 2020.

[21] B. Sischka and G. Kauermann, "EM-based smooth graphon estimation using MCMC and spline-based approaches," *Social Networks*, vol. 68, pp. 279–295, 2022.

[22] S. Chan and E. Airoldi, "A consistent histogram estimator for exchangeable graph models," in *Intl. Conf. on Machine Learning (ICML)*, pp. 208–216, PMLR, 2014.

[23] S. H. Chan, T. B. Costa, and E. M. Airoldi, "Estimation of exchangeable graph models by stochastic blockmodel approximation," in *IEEE Global Conf. Signal and Info. Process. (GlobalSIP)*, pp. 293–296, IEEE, 2013.

[24] K. Pelckmans, J. De Brabanter, J. A. Suykens, and B. De Moor, "Convex clustering shrinkage," in *Statistics and Optimization of Clustering Workshop (PASCAL)*, 2005.

[25] T. D. Hocking, A. Joulin, F. Bach, and J.-P. Vert, "Clusterpath: An algorithm for clustering using convex fusion penalties," in *Intl. Conf. on Machine Learning (ICML)*, p. 1, 2011.

[26] F. Lindsten, H. Ohlsson, and L. Ljung, "Clustering using sum-of-norms regularization: With application to particle filter output computation," in *IEEE Wrkshp. Statistical Signal Process. (SSP)*, pp. 201–204, IEEE, 2011.

[27] H. Ohlsson, L. Ljung, and S. Boyd, "Segmentation of ARX-models using sum-of-norms regularization," *Automatica*, vol. 46, no. 6, pp. 1107–1111, 2010.

[28] K. M. Tan and D. Witten, "Statistical properties of convex clustering," *Electronic Journal of Statistics*, vol. 9, no. 2, p. 2324, 2015.

[29] C. Zhu, H. Xu, C. Leng, and S. Yan, "Convex optimization procedure for clustering: Theoretical revisit," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[30] E. C. Chi and S. Steinerberger, "Recovering trees with convex clustering," *SIAM Journal on Mathematics of Data Science*, vol. 1, no. 3, pp. 383–407, 2019.

[31] H. Choi and S. Lee, "Convex clustering for binary data," *Advances in Data Analysis and Classification*, vol. 13, no. 4, pp. 991–1018, 2019.

[32] M. Navarro, G. I. Allen, and M. Weylandt, "Network clustering for latent state and changepoint detection," *arXiv preprint arXiv:2111.01273*, 2021.

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

[34] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," *arXiv preprint arXiv:1810.00826*, 2018.

[35] C. Morris, N. M. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann, "TUDataset: A collection of benchmark datasets for learning with graphs," *arXiv preprint arXiv:2007.08663*, 2020.