AugWard: Augmentation-Aware Representation Learning for Accurate Graph Classification

Minjun Kim[1], Jaehyeon Choi[1], SeungJoo Lee[1], Jinhong Jung[2⋆], and U Kang[1⋆]

[1] Seoul National University, Seoul, South Korea
{minjun.kim,jaehyeon_choi,hera0131,ukang}@snu.ac.kr
[2] Soongsil University, Seoul, South Korea
jinhong@ssu.ac.kr

Abstract. How can we accurately classify graphs? Graph classification is a pivotal task in data mining with applications in social network analysis, web analysis, drug discovery, molecular property prediction, etc. Graph neural networks have achieved the state-of-the-art performance in graph classification, but they consistently struggle with overfitting. To mitigate overfitting, researchers introduced various representation learning methods utilizing graph augmentation. However, existing methods rely on simplistic use of graph augmentation, which loses augmentation-induced differences and limits the expressiveness of representations.
In this paper, we propose **AugWard** (**Aug**mentation-A**war**e Training with Graph **D**istance and Consistency Regularization), a novel graph representation learning framework that carefully considers the diversity introduced by graph augmentation. AugWard applies augmentation-aware training to predict the graph distance between the augmented graph and its original one, aligning the representation difference directly with graph distance at both feature and structure levels. Furthermore, AugWard employs consistency regularization to encourage the classifier to handle richer representations. Experimental results show that AugWard gives the state-of-the-art performance in supervised, semi-supervised graph classification, and transfer learning.

Keywords: Augmentation-aware Training · Graph Classification · Graph Augmentation · Representation Learning.

1 Introduction

Graph classification [17] is an important data mining task that aims to classify a graph into predefined classes based on its structural properties and features. This task has been attracting attention as it enables reliable predictions from graph-structured data in various applications. Previous studies [12,15] have consistently shown the advantages of Graph Neural Networks (GNNs) [5,9] over traditional methods based on graph kernels. GNNs capture higher-order structures through end-to-end multi-layered architectures, overcoming the limited expressiveness due to hand-crafted features of graph kernels. With the rise of

⋆ Corresponding Authors.

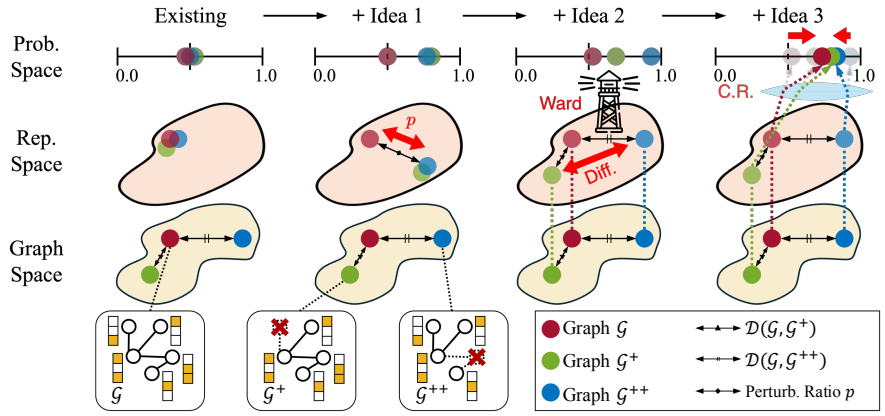


Fig. 1: Main ideas of AugWard are: I1) augmentation-aware training, I2) graph distance-based difference, and I3) Consistency Regularization (C.R.).

GNNs, graph augmentation [11,27] has become crucial to overcome the overfitting issue. Recent approaches further enhance training for better generalization under unsupervised or semi-supervised settings, such as contrastive learning [1,28] or mutual information maximization [19]. Given this background, the efficient use of graph augmentation is a key aspect for graph classification.

However, the state-of-the-art methods [1,10,28] are suboptimal due to two major limitations by simplistic adaptation of graph augmentation. First, the difference between augmented graphs and their originals in terms of structures and features, termed the *augmentation-induced difference*, is largely ignored in existing methods (see Figure 3). They aim to predict consistent labels for augmented graphs and their originals without capturing variations, leading to augmentation-invariant representations. Second, existing methods rely on the deceptive assumption that the perturbation ratio p ensures similarity among augmented graphs. Graph augmentation techniques [11,27] randomly modify graphs according to p (or the extent of change), assuming that augmented graphs with the same p are similar. However, this misleading assumption overlooks the significant diversity among the augmented graphs generated at a fixed p (see Figure 4), limiting previous designs from adequately considering the difference.

We propose **AugWard**, a novel framework for graph representation learning designed to effectively address these limitations. Figure 1 depicts our ideas. We perform augmentation-aware training to enrich graph representations, by aligning the difference between augmented graphs and their original with that estimated by their representations (Idea 1). Observing that the perturbation ratio p is inappropriate to represent graph-level differences, we explicitly measure the graph distance between original and augmented graphs, considering both structure and features with Fused Gromov-Wasserstein distance (Idea 2). Furthermore, we exploit consistency regularization to ensure similar predictions from distinguishable representations of original and augmented graphs (Idea 3).

AugWard is powerful and versatile since it is easily integrated with any method utilizing graph augmentation, enhancing their accuracy across various

learning settings, such as supervised, semi-supervised, and transfer learning. Our main contributions are summarized as follows:

- **Observation.** We empirically observe that existing methods disregard the augmentation-induced difference and an appropriate metric is required to accurately measure this difference (See Figures 3 and 4).
- **Framework.** We propose AugWard, a novel graph representation learning framework that enriches the graph representations by the efficient use of graph augmentation. AugWard incorporates augmentation-aware training with graph distance and consistency regularization (See Figures 1 and 2).
- **Experiments.** AugWard consistently elevates the classification accuracy of the state-of-the-art methods in supervised, semi-supervised graph classification, and transfer learning tasks (see Tables 1, 2, and 3).

The source code of AugWard and the datasets are available at `https://github.com/snudatalab/AugWard`.

2 Preliminaries

Problem Definition. Graph classification predicts the label of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}$ and $\mathcal{E}$ are the sets of nodes and edges, respectively. $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the feature matrix where $n = |\mathcal{V}|$ and d is the number of features.

Problem 1 (Graph Classification).

- **Input**: a set $G = \{\mathcal{G}_1, \cdots, \mathcal{G}_N\}$ of N distinct graphs and a set $Y = \{y_1, \cdots, y_N\}$ of labels where each label y_i corresponds to $\mathcal{G}_i$ and belongs to a set C of classes.
- **Output**: prediction probabilities $P(y|\mathcal{G}_i)$ that each graph $\mathcal{G}_i$ is associated with the label $y \in C$.

Graph Neural Networks. Graph Neural Networks (GNNs) [23,24] are deep learning models designed to process graph-structured data by leveraging message passing algorithm between nodes to learn their representations. They jointly train an encoder f_θ and a classifier g_ϕ for the graph classification task. The encoder f_θ transforms a given graph $\mathcal{G}$ into its representation $\mathbf{z}_\mathcal{G} \in \mathbb{R}^d$ using learnable parameters θ, where d is the embedding dimension. f_θ consists of L layers, where the message-passing step of the l-th layer is as follows:

$$\tilde{\mathbf{h}}_u^{(l)} \leftarrow \texttt{AGGREGATE}\Big(\big\{\mathbf{h}_v^{(l-1)} : v \in \mathcal{N}_u\big\}\Big),\ \mathbf{h}_u^{(l)} \leftarrow \texttt{COMBINE}\Big(\mathbf{h}_u^{(l-1)}, \tilde{\mathbf{h}}_u^{(l)}\Big), \quad (1)$$

where $\mathbf{h}_u^{(l)}$ is the hidden embedding of node u at layer l, and $\mathcal{N}_u$ is the set of neighbors for node u. $\mathbf{h}_u^{(0)}$ is initialized with $\mathbf{x}_u$, the u-th row vector of $\mathbf{X}$. After L propagation, $\mathbf{z}_\mathcal{G}$ is obtained as follows, where $[L] \coloneqq \{0, \cdots, L\}$:

$$\mathbf{z}_\mathcal{G} \leftarrow \texttt{READOUT}\Big(\big\{\mathbf{h}_u^{(l)} \mid u \in \mathcal{V}, l \in [L]\big\}\Big). \quad (2)$$

Given $\mathbf{z}_\mathcal{G}$, the classifier g_ϕ produces a vector $\mathbf{p}_\mathcal{G} \in \mathbb{R}^{|C|}$ of predicted probabilities, i.e., $\mathbf{p}_\mathcal{G} \leftarrow g_\phi(\mathbf{z}_\mathcal{G})$, where the i-th entry of $\mathbf{p}_\mathcal{G}$ is $P(y = i|\mathcal{G})$. In this paper, we assume that class labels are represented as integers between 1 and $|C|$, i.e., $P(y = i|\mathcal{G}_G)$ indicates the probability that label y of $\mathcal{G}_G$ belongs to class i.

Graph Augmentation. Graph augmentation aims to increase the volume of graph instances by modifying the original graphs while retaining their labels.

Let $\mathcal{T}_p(\cdot|\mathcal{G})$ represent the augmentation distribution conditioned on the original graph $\mathcal{G}$, where the hyperparameter p denotes the perturbation ratio indicating the amount of change from $\mathcal{G}$. Then, an augmented graph $\mathcal{G}^+$ is randomly sampled as $\mathcal{G}^+ \sim \mathcal{T}_p(\mathcal{G}^+ \mid \mathcal{G})$. Various designs have been proposed for $\mathcal{T}_p$, such as drop-based methods [21,28] or mixup-based methods [13,15]. The former randomly either remove or mask attributes of nodes and edges according to the ratio p. The latter fuse two source graphs by the ratio p, with the resulting graph having a linearly interpolated label from the original graphs.

Fused Gromov-Wasserstein Distance. Fused Gromov-Wasserstein Distance (FGWD) [13,20] measures the distance between two graphs by combining both feature-level and structure-level differences, inspired from the optimal transport problem. FGWD employs Wasserstein distance $\texttt{WD}(\cdot,\cdot,\cdot)$ to capture feature-level differences and Gromov-Wasserstein distance $\texttt{GWD}(\cdot,\cdot,\cdot)$ for structural variations. FGWD between two graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ and $\mathcal{G}^+ = (\mathcal{V}^+, \mathcal{E}^+, \mathbf{X}^+)$ is as follows:

$$\texttt{FGWD}_\alpha(\mathcal{G}, \mathcal{G}^+) \coloneqq \min_{\boldsymbol{\pi} \in \Pi(\boldsymbol{\mu}, \boldsymbol{\nu})} \alpha \cdot \texttt{WD}(\mathbf{X}, \mathbf{X}^+, \boldsymbol{\pi}) + (1-\alpha) \cdot \texttt{GWD}(\mathcal{E}, \mathcal{E}^+, \boldsymbol{\pi}),$$

where α is a balancing hyperparameter, $\boldsymbol{\pi}$ is the coupling matrix [20] describing the probabilistic matching between two graphs, $\Pi(\boldsymbol{\mu}, \boldsymbol{\nu})$ is the set of all coupling matrices that align $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$, $\boldsymbol{\mu} \in \mathbb{R}^n$ and $\boldsymbol{\nu} \in \mathbb{R}^{n^+}$ are the source and target distributions on $\mathcal{V}$ and $\mathcal{V}^+$, respectively. There are traditional metrics for measuring graph distance, but most of them focus only on structural differences, with a few heuristically considering features. In contrast, FGWD 1) flexibly balances both differences by α within a single optimization, 2) is computationally tractable compared to NP-hardness of graph edit distance [16], and 3) provides precise distances by following the optimal transport setting.

3 Proposed Method

We propose **AugWard**, a novel graph representation learning framework for accurate graph classification. The technical challenges in improving the performance of graph classification with augmented graphs are as follows:

C1. Capturing augmentation-induced difference. Previous approaches encourage the encoder to generate representations invariant to graph augmentation, thereby disregarding the difference between the augmented graph and its original. How can we train the model to capture such a difference?

C2. Measuring the difference gained from graph augmentation. Augmentation techniques rely on the perturbation ratio to control the extent of changes. However, this ratio is not suitable for representing the actual difference due to randomness. How can we measure this difference?

C3. Training robust classifier. Existing approaches produce augmentation-invariant representations, which limit the classifier's ability to handle the diversity from graph augmentations. How can we train a robust classifier to better generalize and fully utilize expressive graph representations?

We propose the following ideas to deal with the challenges above:

I1. Augmentation-aware training (Sec. 3.1). We propose a model-agnostic learning strategy that aligns the difference between the augmented graph

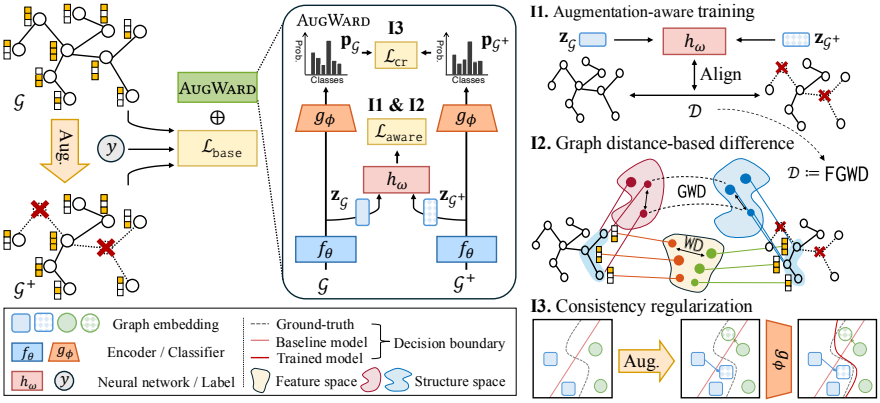


Fig. 2: Overall process of AugWard. AugWard incorporates augmentation-aware loss $\mathcal{L}_{\texttt{aware}}$ (**I1** & **I2**) and consistency regularization loss $\mathcal{L}_{\texttt{cr}}$ (**I3**).

and its original with that of their representations, enabling the training model to be aware of the augmentation-induced difference.

I2. Graph distance-based difference (Sec. 3.2). We measure the graph distance between the augmented graph and its original as their graph-level difference. Specifically, we exploit FGWD since it captures the variations in both features and structures induced by graph augmentation.

I3. Consistency regularization (Sec. 3.3). We jointly optimize consistency regularization term that aligns the predictions of an augmented graph with those of its original graph, fully harnessing expressive representations obtained through our augmentation-aware training.

Figure 2 depicts the overall procedure of AugWard. Our framework jointly optimizes $\mathcal{L}_{\texttt{aware}}$ and $\mathcal{L}_{\texttt{cr}}$ along with a baseline loss $\mathcal{L}_{\texttt{base}}$ (e.g., semi-supervised learning), where any technique is adoptable for $\mathcal{T}_p$, f_θ, and g_ϕ.

3.1 Augmentation-aware Training

Observation. We first present an empirical observation that explains why the existing approaches fail to reflect the difference induced by augmentation in the representations between the augmented graph $\mathcal{G}^+$ and its original $\mathcal{G}$. We assume that ideal representations are able to distinguish graph-level differences, and thus their representations should differ in proportion to how much the augmented graph deviates from the original graph, i.e., there should be a positive correlation between them. To check this, we begin by applying augmentations of varying intensities to a graph $\mathcal{G}$ sampled from the `PROTEINS` [14] dataset. For each perturbation ratio $p \in \{0.05, 0.1, \cdots, 0.45\}$, we generate 100 augmented graphs $\mathcal{G}^+$. After training a GIN encoder, we measure the Euclidean distance $\|\mathbf{z}_\mathcal{G} - \mathbf{z}_{\mathcal{G}^+}\|_2^2$ of their representations and the actual graph distance (in FGWD) between $\mathcal{G}$ and $\mathcal{G}^+$. As shown in Figure 3, Pearson Correlation Coefficients (PCC) are almost zero, indicating that there is no strong correlation between representation-level and graph-level differences. This clearly verifies the loss of augmentation-induced differences in existing representation learning methods.

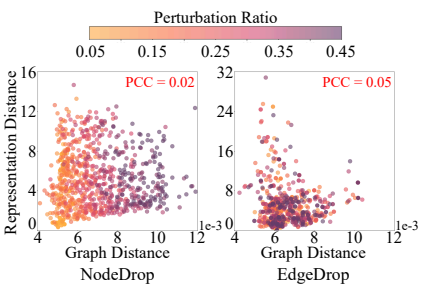


Fig. 3: The representation-level and graph-level differences between $\mathcal{G}$ and $\mathcal{G}^+$ show no correlation. See Section 3.1 for details.

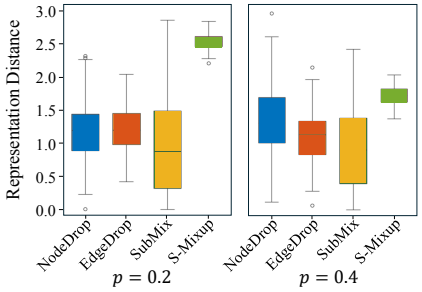


Fig. 4: Graphs with the same perturbation ratio p demonstrate notable diversity across various augmentations. See Section 3.2 for details.

Solution. Motivated by this observation, we aim to encourage the encoder f_θ to align the representation-level difference with the graph-level difference. For this purpose, we train a neural network h_ω with parameters ω using the graph representations $\mathbf{z}_\mathcal{G}$ and $\mathbf{z}_{\mathcal{G}^+}$ to capture the difference between the original and augmented graphs $\mathcal{G}$ and $\mathcal{G}^+$. This is achieved by optimizing $\mathcal{L}_{\texttt{aware}}$:

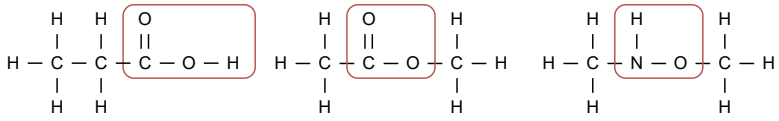
$$\mathcal{L}_{\texttt{aware}}(\mathcal{G}, \mathcal{G}^+) \coloneqq \left\| h_\omega(\mathbf{z}_\mathcal{G}, \mathbf{z}_{\mathcal{G}^+}) - \mathcal{D}(\mathcal{G}, \mathcal{G}^+) \right\|_2^2, \tag{3}$$

where $\mathcal{D}(\mathcal{G}, \mathcal{G}^+)$ denotes the difference between the graphs, and we employ a fully connected layer h_ω with the concatenation of $\mathbf{z}_\mathcal{G}$ and $\mathbf{z}_{\mathcal{G}^+}$ as input for h_ω. The loss injects the graph-level difference $\mathcal{D}$ into the relationship between the representations, captured by the neural network h_ω, so that the difference information is encoded into the representations. Note that there would be various options for $\mathcal{D}$, and any choice would be adopted within the loss function. Which is suitable for measuring the difference, especially in the context of graph classification? We present a detailed discussion on our design choice of $\mathcal{D}$ in Section 3.2.

3.2 Graph Distance-based Difference

Observation. A naïve answer for $\mathcal{D}(\mathcal{G}, \mathcal{G}^+)$ is the perturbation ratio p, as it indicates the amount of change from the original graph through graph augmentation $\mathcal{T}_p$. However, this approach fails to precisely represent the difference between $\mathcal{G}$ and $\mathcal{G}^+$, because generated graphs at a fixed perturbation ratio p exhibit significant variations due to inherent randomness. To investigate this, we generate 100 augmented graphs $\mathcal{G}^+$ from a sampled graph $\mathcal{G}$ from the `PROTEINS` dataset for each augmentation method $\mathcal{T}_p$ with p fixed at 0.2 and 0.4, respectively. Then, we measure the Euclidean distances $\|\mathbf{z}_\mathcal{G} - \mathbf{z}_{\mathcal{G}^+}\|_2^2$ (i.e. $\mathbf{z}_\mathcal{G} = f_\theta(\mathcal{G})$, $\mathbf{z}_{\mathcal{G}^+} = f_\theta(\mathcal{G}^+)$). Figure 4 illustrates the distributions of the representation-level distances for each p and augmentation method. These distributions show significant variance and inconsistency across different augmentation types and ratios. Hence, the perturbation ratio p is inappropriate to represent the difference $\mathcal{D}$.

Solution. Our idea is to employ a graph distance metric to explicitly measure the difference between $\mathcal{G}$ and $\mathcal{G}^+$. Specifically, we exploit Fused Gromov-Wasserstein Dimstance (FGWD) for this purpose. The main reason is as follows.



(a) $C_3H_6O_2$ (Acid) (b) $C_3H_6O_2$ (Neutral) (c) C_2H_7NO (Base)

Fig. 5: Graphs of three compounds. (a) and (b) share graph features, while (b) and (c) exhibit identical graph structures. Functional groups are marked in red.

In the graph classification task, each graph $\mathcal{G}$ includes its topological structure, defined by $\mathcal{V}$ and $\mathcal{E}$, along with node features $\mathbf{X}$. Graph augmentation alters either structure or features, which are effectively captured by FGWD. Figure 5 shows examples of chemical compound graphs, showing why considering both structure and features is important. In these graphs, (a) and (b) share identical features, while (b) and (c) exhibit the same structure. The difference in feature or structure, leads to a distinction in chemical types. Therefore, it is desirable to consider differences in both structure and features.

Although there are traditional graph distance metrics, most consider only structural information or rely on heuristics to consider features. In contrast, FGWD effectively captures both aspects, showing advantages as discussed in Section 2. Thus, we measure FGWD as the difference $\mathcal{D}$ between $\mathcal{G}$ and $\mathcal{G}^+$. The loss function of our augmentation-aware training is represented as follows:

$$\mathcal{L}_{\texttt{aware}}(\mathcal{G}, \mathcal{G}^+) \coloneqq \left\| h_\omega(\mathbf{z}_\mathcal{G}, \mathbf{z}_{\mathcal{G}^+}) - \texttt{FGWD}_\alpha(\mathcal{G}, \mathcal{G}^+) \right\|_2^2, \tag{4}$$

where α is the hyperparameter of FGWD. We set $\boldsymbol{\mu} = \mathbf{1}_n/n$ and $\boldsymbol{\nu} = \mathbf{1}_{n^+}/n^+$ following [13], where $n = |\mathcal{V}|$, $n^+ = |\mathcal{V}^+|$, and $\mathbf{1}_n$ is an all-one vector of size n.

3.3 Consistency Regularization

Motivation. Given distinguishable representations, it is important to train the classifier robustly for better generalization. In node classification, GRAND [2] points out that matching predictions from different representations for the same label improves the model's generalization behavior. Inspired by this, we design a loss for consistency regularization for graph classification, so that the classifier g_ϕ fully utilizes the expressive representations by augmentation-aware training.

Solution. As shown in Figure 2, the classifier g_ϕ yields $\mathbf{p}_\mathcal{G}$ and $\mathbf{p}_{\mathcal{G}^+}$, vectors of prediction scores, from $\mathbf{z}_\mathcal{G}$ and $\mathbf{z}_{\mathcal{G}^+}$. The idea of the consistency regularization is to match the two predictions $\mathbf{p}_\mathcal{G}$ and $\mathbf{p}_{\mathcal{G}^+}$, with the loss represented as follows:

$$\mathcal{L}_{\texttt{cr}}(\mathcal{G}, \mathcal{G}^+) \coloneqq H(\mathbf{p}_\mathcal{G}, \mathbf{p}_{\mathcal{G}^+}) = -\sum_{i=1}^{|C|} P(y = i|\mathcal{G}) \cdot \log P(y = i|\mathcal{G}^+), \tag{5}$$

where H denotes the cross-entropy, and the i-th entry of $\mathbf{p}_\mathcal{G}$ indicates $P(y = i|\mathcal{G})$ (similar for $\mathcal{G}^+$). In other words, it regulates the classifier to make consistent predictions for both $\mathcal{G}$ and $\mathcal{G}^+$, aiming to improve robustness and accuracy.

3.4 Final Loss Function

We sum up all the loss functions for AugWard as follows:

$$\mathcal{L}_{\texttt{AugWard}}(\mathcal{G}, \mathcal{G}^+) \coloneqq \lambda_{\texttt{aware}} \cdot \mathcal{L}_{\texttt{aware}}(\mathcal{G}, \mathcal{G}^+) + \lambda_{\texttt{cr}} \cdot \mathcal{L}_{\texttt{cr}}(\mathcal{G}, \mathcal{G}^+), \tag{6}$$

where $\lambda_{\texttt{aware}}$ and $\lambda_{\texttt{cr}}$ are hyperparameters that control the strength of their respective losses. Our framework supports various baseline learning paradigms

Table 1: Accuracy of supervised graph classification for various graph augmentation methods, where "Imp." indicates the percentage point improvement of the average accuracy when AugWard is applied. AugWard consistently enhances the performance with those augmentation methods on various datasets.

Models	DD	ENZ*	I-B*	I-M*	NCI1	NCI109	PTC*	PRO*	R-B*	TWI*	Avg.	Imp.
NodeDrop [28]	75.42	27.67	56.40	44.53	67.15	65.04	67.24	70.95	76.30	65.14	61.58	-
+ AugWard	76.69	28.67	58.40	45.87	67.79	65.52	76.44	72.76	78.40	65.30	**63.58**	**+2.00**
EdgeDrop [28]	71.39	26.67	55.20	46.13	66.13	64.79	68.94	67.39	77.10	65.16	60.89	-
+ AugWard	75.84	29.00	60.60	46.93	67.10	65.08	75.29	67.20	77.50	65.61	**63.02**	**+2.13**
AttrMask [28]	69.15	26.67	56.60	46.93	65.84	64.60	68.37	67.44	74.50	65.10	60.52	-
+ AugWard	72.54	29.33	58.80	48.40	67.06	64.94	73.01	67.92	78.30	65.50	**62.58**	**+2.06**
Subgraph [28]	71.02	25.00	56.40	46.53	64.40	64.67	69.56	64.34	75.30	64.89	60.21	-
+ AugWard	72.71	25.33	60.40	45.20	65.57	65.11	72.39	69.35	75.10	65.08	**61.62**	**+1.41**
GraphAug [12]	71.53	27.67	57.40	45.92	64.96	61.26	68.35	69.52	77.20	65.05	60.89	-
+ AugWard	73.90	28.00	60.60	46.77	66.96	62.28	70.10	70.54	78.20	65.21	**62.26**	**+1.37**
SubMix [27]	79.83	27.00	59.40	45.11	64.70	63.83	70.67	68.63	76.40	64.67	62.02	-
+ AugWard	80.00	27.67	60.80	45.07	65.06	64.02	71.28	68.83	79.00	65.37	**62.71**	**+0.69**
S-Mixup [11]	79.29	27.33	58.60	46.67	65.11	64.26	68.35	62.54	77.60	64.96	61.47	-
+ AugWard	79.49	26.33	59.80	46.80	65.50	64.60	70.67	65.69	76.80	65.34	**62.10**	**+0.63**

* ENZ: ENZYMES, I-B: IMDB-BINARY, I-M: IMDB-MULTI, PTC: PTC-MR, PRO: PROTEINS, R-B: REDDIT-BINARY, TWI: TWITTER

including supervised, semi-supervised, and transfer learning for graph classification. This is achieved by jointly optimizing $\mathcal{L}_{\texttt{base}}$ and $\mathcal{L}_{\texttt{AugWard}}$ as follows:

$$\mathcal{L}(\mathcal{G}, \mathcal{G}^+, y) \coloneqq \mathcal{L}_{\texttt{base}}(\mathcal{G}, \mathcal{G}^+, y) + \mathcal{L}_{\texttt{AugWard}}(\mathcal{G}, \mathcal{G}^+), \tag{7}$$

where $\mathcal{L}_{\texttt{base}}(\mathcal{G}, \mathcal{G}^+, y)$ is the baseline loss with the ground-truth label y (e.g., the sum of cross-entropy losses for $\mathcal{G}$ and $\mathcal{G}^+$ in supervised learning).

4 Experiments

We perform experiments to answer the following questions:

Q1. Accuracy in supervised graph classification (Section 4.1). Does AugWard improve the graph classification accuracy of supervised classification methods utilizing graph augmentation techniques?

Q2. Accuracy in semi-supervised graph classification (Section 4.2). Does AugWard improve the graph classification accuracy of semi-supervised classification methods utilizing graph augmentation techniques?

Q3. Representation transferability (Section 4.3). How accurate is AugWard in transfer learning? Does it enhance representation transferability?

Q4. Runtime analysis (Section 4.4). How significant is the additional computational overhead caused by AugWard?

Q5. Ablation study (Section 4.5). Are all the components of AugWard effective in improving model performance?

Experimental Setup. We evaluate AugWard in various settings, including supervised, semi-supervised, and transfer learning. Following [23,27], we use ten benchmark datasets [14] for supervised and semi-supervised learning, where the label rate is 10% for the semi-supervised setting. For transfer learning, we evaluate AugWard with ZINC15 [18] as the source and eight downstream datasets from MoleculeNet [22] as the target [4]. We set the learning rate to 0.01 for the Adam optimizer and train a four-layered GIN for 100 epochs. We conduct a grid

Table 2: Accuracy [%] of semi-supervised graph classification using NodeDrop as graph augmentation within each baseline, where only 10% labeled graphs are available for training. AugWard provides improved performance in average accuracy, verifying its effectiveness also in the semi-supervised setting.

Models	DD	ENZ*	I-B*	I-M*	NCI1	NCI109	PTC*	PRO*	R-B*	TWI*	Avg.	Imp.
InfoGraph [19]	63.64	20.67	50.20	37.33	61.68	61.91	64.91	65.39	66.20	55.29	54.72	-
+ AugWard	63.98	22.67	54.60	38.79	62.89	62.44	66.08	66.29	67.50	57.18	**56.24**	**+1.52**
GraphCL [28]	63.39	21.00	53.80	39.47	61.92	61.19	66.66	64.37	67.10	56.71	55.56	-
+ AugWard	64.18	23.67	55.20	39.73	63.50	62.83	67.24	65.93	68.20	57.09	**56.76**	**+1.20**
CuCo [1]	63.81	23.33	52.40	39.27	61.97	60.40	65.61	65.57	67.40	57.11	55.69	-
+ AugWard	64.66	24.00	54.20	39.80	62.17	61.21	68.78	67.39	68.00	57.28	**56.75**	**+1.06**
GCL-SPAN [10]	63.22	23.67	53.20	40.20	62.96	61.46	65.64	66.84	67.50	57.73	56.24	-
+ AugWard	64.26	24.33	55.80	41.26	64.47	62.05	67.39	67.80	68.30	57.92	**57.36**	**+1.12**

* ENZ: ENZYMES, I-B: IMDB-BINARY, I-M: IMDB-MULTI, PTC: PTC-MR, PRO: PROTEINS, R-B: REDDIT-BINARY, TWI: TWITTER

Table 3: ROC-AUC of transfer learning experiments, pretrained on ZINC15 [18] and fine-tuned on MoleculeNet [22] datasets. AugWard improves existing models for transfer learning, offering more expressive graph representations.

Models	BACE	BBBP	ClinTox	HIV	MUV	Tox21	ToxCast	SIDER	Avg.	Imp.
ContextPred [4]	80.80	71.76	70.22	77.64	76.14	75.56	63.00	61.43	72.07	-
+ AugWard	83.32	72.85	70.60	79.35	80.80	75.81	63.94	62.82	**73.69**	**+1.62**
GraphCL [28]	73.15	70.33	73.80	80.08	69.30	74.44	62.32	60.48	70.49	-
+ AugWard	77.24	72.78	82.41	81.42	77.67	75.69	63.46	61.66	**74.04**	**+3.55**
MGSSL [29]	80.83	72.76	76.74	73.30	72.35	74.87	62.02	56.07	71.12	-
+ AugWard	85.41	74.26	82.15	77.95	77.20	76.09	63.81	61.77	**74.83**	**+3.71**
GraphMAE [3]	81.30	72.04	82.82	77.15	72.21	75.33	64.07	61.07	73.25	-
+ AugWard	83.50	73.19	84.88	78.63	81.86	75.76	64.08	61.62	**75.44**	**+2.19**

search of hyperparameters: batch size $\in \{32, 128\}$, dropout $\in \{0, 0.5\}$, $p \in \{0.05, 0.1, 0.15, 0.2\}$, $\alpha \in \{0.05, 0.5, 0.95\}$, $\lambda_{\texttt{aware}} \in \{5, 10, 25, 50, 75, 100\}$, and $\lambda_{\texttt{cr}} \in \{0, 0.1, 1, 10, 100\}$. All experiments are done on a single RTX 3090 GPU.

4.1 Supervised Graph Classification (Q1)

We examine the effect of AugWard for various augmentations on supervised graph classification. We train with each augmentation method as baseline and compare with AugWard. Table 1 shows the results on various datasets, and reports the average accuracy ("Average") and the percentage point improvement ("Imp."). AugWard consistently improves accuracy for all tested augmentation methods across most datasets, with increases in average accuracy of up to 2.13%p. This indicates that AugWard effectively improves learning performance, while being easily integrated.

4.2 Semi-supervised Graph Classification (Q2)

We evaluate the effectiveness of AugWard for Semi-Supervised Learning (SSL) methods on graph classification, where labels are available for 10% of the graphs. We choose NodeDrop [28] as the default augmentation for each SSL method. Table 2 shows AugWard enhances the SSL models across various datasets, achieving up to 1.52%p increase in average accuracy. The result verifies that our strategy is beneficial also for SSL, as well as supervised learning.

4.3 Representation Transferability (Q3)

We investigate how AugWard improves the transferability of graph representations in transfer learning models. For this, we pretrain a model on the

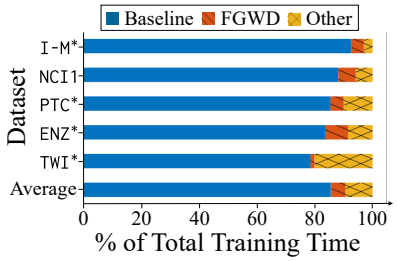


Fig. 6: Proportion of running time in AugWard components.

Table 4: Ablation study for our proposed ideas in AugWard. All ideas of AugWard enhance the performance.

Variants	Ideas	Avg.	Imp.
A: GIN + NodeDrop	Existing	61.58	-
A + p	I1	61.90	+0.32
A + NFs	I1	62.50	+0.92
A + AMs	I1	62.55	+0.98
A + Edge Jaccard	I1	62.53	+0.95
A + FGWD	I1+I2	63.04	+1.46
A + **AugWard**	I1+I2+I3	**63.58**	**+2.00**

ZINC15 [18] dataset and fine-tune it on eight downstream datasets from MoleculeNet [22]. As shown in Table 3, AugWard consistently improves the performance of different transfer learning models for most downstream tasks in terms of ROC-AUC, specifically up to 3.71%p higher average accuracy. This indicates that AugWard produces enhanced representations for transfer learning.

4.4 Runtime Analysis (Q4)

We analyze the computational overhead of AugWard. For this, we measure the training time of AugWard, which consists of 1) baseline supervised learning with augmentation, 2) computing FGWD, and 3) other remaining parts. Figure 6 shows the proportion of each component over the total training time for five datasets of various sizes. Note that the overhead from FGWD is marginal, i.e., computing FGWD takes 4.89% of the total time in average. Thus, AugWard achieves a favorable trade-off between time and accuracy.

4.5 Ablation Study (Q5)

To evaluate the effectiveness of our proposed ideas, we conduct an ablation study by incorporating each idea incrementally. Starting with GIN and NodeDrop as the baseline, we examine four variants for augmentation-aware training (Idea 1), where the augmentation-induced difference $\mathcal{D}$ formulated as: (1) perturbation ratio p, (2) difference of Node Features (NFs), (3) difference of Adjacency Matrices (AMs), and (4) edge Jaccard similarity. We further integrate graph distance-based difference (Idea 2) using FGWD and finally apply consistency regularization (Idea 3) to demonstrate the cumulative benefit of all three ideas. Table 4 reports the average classification accuracy across ten datasets where we have four observations. First, considering the difference is beneficial, even though they are simple heuristics. Second, p is less effective than other variants. Third, selecting FGWD as $\mathcal{D}$ shows the best performance among different metrics. Last, the highest accuracy is achieved through the joint application of all the three ideas. Overall, all our proposed ideas contribute to the enhanced performance.

5 Related Work

Graph representation learning. Learning graph representation [7,8] plays a crucial role in classifying graphs, enabling models to make predictions on graphs [25,26]. Previous studies have shown that GNN-based methods [3,10] perform better than similarity-based approaches such as graph kernels [6]. These

methods employ GNNs to capture higher-order structures through multi-layered message-passing, and yield representations via pooling. For better generalization, recent studies exploit augmentation and further train with advanced strategies such as mutual information maximization [19] and contrastive learning [1,28].

Learning graphs with augmentation. Graph augmentation produces a new set of graphs from an original set, ensuring the new graphs share similar characteristics with their originals. There are three categories of graph augmentation: structure-oriented, feature-oriented, and mixup-based methods. Structure-oriented techniques [12,21] modify graph structures by randomly dropping nodes and edges or rewiring nodes. Feature-oriented approaches [28] alter node or edge features by randomly masking or shuffling them. Mixup-based methods [13,27] create new graphs by interpolating between pairs of graphs. However, these methods focus on augmentation-invariant representation, thus fail to capture the diversity between graphs in their representations. AugWard effectively captures augmentation-induced differences and improves model performance.

6 Conclusion

We propose AugWard, a novel graph representation learning framework that considers augmentation-induced differences for accurate graph classification. Our main idea is augmentation-aware training with graph distance and consistency regularization for improving both the quality of the graph representations and prediction accuracy. Experimental results show that AugWard is robust and adaptable, improving performance in diverse settings such as augmentation, mixup, supervised, semi-supervised classification, and transfer learning. Future works include extending AugWard for classifying other types of graphs.

Acknowledgments. This work was supported by the National Research Foundation of Korea(NRF) funded by MSIT(2022R1A2C3007921). This work was also supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) [No.2022-0-00641, XVoice: Multi-Modal Voice Meta Learning], [No.RS-2024-00509257, Global AI Frontier Lab], [No.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University)], and [NO.RS-2021-II212068, Artificial Intelligence Innovation Hub]. The Institute of Engineering Research at Seoul National University provided research facilities for this work. The ICT at Seoul National University provides research facilities for this study. U Kang and Jinhong Jung are the corresponding authors.

References

1. Chu, G., Wang, X., Shi, C., Jiang, X.: Cuco: Graph representation with curriculum contrastive learning. In: IJCAI (2021)
2. Feng, W., Zhang, J., Dong, Y., Han, Y., Luan, H., Xu, Q., Yang, Q., Kharlamov, E., Tang, J.: Graph random neural networks for semi-supervised learning on graphs. In: NeurIPS (2020)
3. Hou, Z., Liu, X., Cen, Y., Dong, Y., Yang, H., Wang, C., Tang, J.: Graphmae: Self-supervised masked graph autoencoders. In: KDD (2022)
4. Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V.S., Leskovec, J.: Strategies for pre-training graph neural networks. In: ICLR (2020)
5. Jung, J., Yoo, J., Kang, U.: Signed random walk diffusion for effective representation learning in signed graphs. PLOS ONE **17**(3), 1–19 (03 2022)

6. Kang, U., Tong, H., Sun, J.: Fast random walk graph kernel. In: SDM (2012)
7. Kim, J., Park, K.H., Yoon, H., Kang, U.: Accurate link prediction for edge-incomplete graphs via pu learning. In: AAAI (2025)
8. Kim, J., Park, H., Lee, J., Kang, U.: SIDE: representation learning in signed directed networks. In: WWW (2018)
9. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
10. Lin, L., Chen, J., Wang, H.: Spectral augmentation for self-supervised learning on graphs. In: ICLR (2023)
11. Ling, H., Jiang, Z., Liu, M., Ji, S., Zou, N.: Graph mixup with soft alignments. In: ICML (2023)
12. Luo, Y., McThrow, M., Au, W.Y., Komikado, T., Uchino, K., Maruhashi, K., Ji, S.: Automated data augmentations for graph classification. In: ICLR (2023)
13. Ma, X., Chu, X., Wang, Y., Lin, Y., Zhao, J., Ma, L., Zhu, W.: Fused gromov-wasserstein graph mixup for graph-level classifications. In: NeurIPS (2023)
14. Morris, C., Kriege, N.M., Bause, F., Kersting, K., Mutzel, P., Neumann, M.: Tu-dataset: A collection of benchmark datasets for learning with graphs. In: ICML Workshop GRL+ (2020)
15. Park, J., Shim, H., Yang, E.: Graph transplant: Node saliency-guided graph mixup with local structure preservation. In: AAAI (2022)
16. Riesen, K.: Structural pattern recognition with graph edit distance. Advances in computer vision and pattern recognition pp. 1–164 (2015)
17. Shim, S., Kim, J., Park, K., Kang, U.: Accurate graph classification via two-staged contrastive curriculum learning. PLOS ONE **19**(1), 1–20 (01 2024)
18. Sterling, T., Irwin, J.J.: ZINC 15 - ligand discovery for everyone. J. Chem. Inf. Model. **55**(11), 2324–2337 (2015)
19. Sun, F., Hoffmann, J., Verma, V., Tang, J.: Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In: ICLR (2020)
20. Vayer, T., Courty, N., Tavenard, R., Chapel, L., Flamary, R.: Optimal transport for structured data with application on graphs. In: ICML (2019)
21. Wang, Y., Wang, W., Liang, Y., Cai, Y., Liu, J., Hooi, B.: Nodeaug: Semi-supervised node classification with data augmentation. In: KDD (2020)
22. Wu, Z., Ramsundar, B., Feinberg, E.N., Gomes, J., Geniesse, C., Pappu, A.S., Leswing, K., Pande, V.: Moleculenet: a benchmark for molecular machine learning. Chemical science **9**(2), 513–530 (2018)
23. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: ICLR (2019)
24. Yoo, J., Jeon, H., Kang, U.: Belief propagation network for hard inductive semi-supervised learning. In: IJCAI (2019)
25. Yoo, J., Kim, J., Yoon, H., Kim, G., Jang, C., Kang, U.: Accurate graph-based PU learning without class prior. In: ICDM (2021)
26. Yoo, J., Kim, J., Yoon, H., Kim, G., Jang, C., Kang, U.: Graph-based PU learning for binary and multiclass classification without class prior. Knowl. Inf. Syst. **64**(8), 2141–2169 (2022)
27. Yoo, J., Shim, S., Kang, U.: Model-agnostic augmentation for accurate graph classification. In: WWW (2022)
28. You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., Shen, Y.: Graph contrastive learning with augmentations. In: NeurIPS (2020)
29. Zhang, Z., Liu, Q., Wang, H., Lu, C., Lee, C.: Motif-based graph self-supervised learning for molecular property prediction. In: NeurIPS (2021)