UMIX: Improving Importance Weighting for Subpopulation Shift via Uncertainty-Aware Mixup

Anonymous Author(s)
Affiliation
Address
email

Abstract

Subpopulation shift wildly exists in many real-world machine learning applications, referring to the training and test distributions containing the same subpopulation groups but varying in subpopulation frequencies. Importance reweighting is a normal way to handle the subpopulation shift issue by imposing constant or adaptive sampling weights on each sample in the training dataset. However, some recent studies have recognized that most of these approaches fail to improve the performance over empirical risk minimization especially when applied to over-parameterized neural networks. In this work, we propose a simple yet practical framework, called uncertainty-aware mixup (UMIX), to mitigate the overfitting issue in over-parameterized models by reweighting the "mixed" samples according to the sample uncertainty. The training-trajectories-based uncertainty estimation is equipped in the proposed UMIX for each sample to flexibly characterize the subpopulation distribution. We also provide insightful theoretical analysis to verify that UMIX achieves better generalization bounds over prior works. Further, we conduct extensive empirical studies across a wide range of tasks to validate the effectiveness of our method both qualitatively and quantitatively.

1 Introduction

Empirical risk minimization (ERM) typically faces challenges from distribution shift, which refers to the difference between training and test distributions [53, 22, 3]. One common type of distribution shift is subpopulation shift wherein the training and test distributions consist of the same subpopulation groups but differ in subpopulation frequencies [6, 8]. Many practical research problems (e.g., fairness of machine learning and class imbalance) can all be considered as a special case of subpopulation shift [25, 16, 23]. For example, in the setting of fair machine learning, we train the model on a training dataset with biased demographic subpopulations and test it on an unbiased test dataset [25, 16]. Therefore the essential goal of fair machine learning is to mitigate the subpopulation shift between training and test datasets.

Many approaches have been proposed for solving this problem. Among these approaches, importance weighting (IW) is a classical yet effective technique by imposing static or adaptive weights on each sample when building weighted empirical loss. Therefore each subpopulation group contributes comparably to the final training objective. Specifically, there are normally two ways to achieve importance re-weighting. Early works propose to re-weight the sample inverse proportionally to the subpopulation frequencies (i.e., static weights) [53, 51, 12, 50, 11, 35], such as class-imbalanced learning approaches [12, 11, 35]. Alternatively, a more flexible way is to re-weight individual samples adaptively according to training dynamics [57, 64, 39, 62, 28, 40, 33, 54]. Distributional robust optimization (DRO) is one of the most representative methods in this line, which minimizes the loss over the worst-case distribution in a neighborhood of the empirical training distribution. A

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

commonly used dual form of DRO can be seen as a special case of importance re-weighting wherein the sampling weights are updated based on the current loss [44, 19, 31, 20] in an alternated manner.

However, some recent studies have shown both empirically and theoretically that these IW methods could fail to achieve better worst-case subpopulation performance compared with ERM. Empirically, prior works [10, 50] recognize that various IW methods tend to exacerbate overfitting, which leads to a diminishing effect on stochastic gradient descent (SGD) over training epochs especially when they are applied to over-parameterized neural networks (NNs). Theoretically, previous studies prove that for over-parameterized neural networks, re-weighting algorithms do not improve over ERM because their implicit biases are (almost) equivalent [63, 51, 59]. In addition, some prior works also point out that using conventional regularization techniques such as weight decay cannot significantly improve the performance of IW [50].

To this end, we introduce a novel technique called uncertainty-aware mixup (UMIX), by re-weighting the mixed samples according to uncertainty within the mini-batch while mitigating overfitting. Specifically, we employ the well-known mixup technique to produce "mixed" augmented samples. Then we train the model on these mixed samples to make sure it can always see "novel" samples thus the effects of IW will not dissipate even at the end of the training epoch. To enforce the model to perform fairly well on all subpopulations, we further efficiently re-weight the mixed samples according to uncertainty of the original samples. The weighted mixup loss function is induced by combining the weighted losses of the corresponding two original samples. At a high level, this approach augments training samples in an uncertainty-aware manner, i.e., putting more focus on samples with higher prediction uncertainties that belong to minority subpopulations with high probabilities. We also show UMIX can provide additional theoretical benefit which achieves a tighter generalization bound than weighted ERM [34, 33, 62, 31]. In summary, the contributionns of this paper are:

- We propose a simple and practical approach called uncertainty-aware mixup (UMIX) to improve previous IW methods by re-weighting the mixed samples, which provides a new framework to mitigate overfitting in over-parameterized neural networks.
- Under the proposed framework, we provide theoretical analysis with insight that UMIX can achieve a tighter generalization bound than the weighted ERM.
- We perform extensive experiments on a wide range of tasks, where the proposed UMIX achieves excellent performance in both group-oblivious and group-aware settings.

Comparison with existing works. Here, we discuss the key differences between UMIX and other works. In contrast to most IW methods (e.g., CVaR-DRO [31] and JTT [34]), UMIX employs a mixup strategy to improve previous IW methods and mitigate the model overfitting. Among these methods, JTT [34] and LISA [61] are the two most related works to ours. Specifically, JTT provides a two-stage optimization framework in which an additional network is used for building the error set, and then JTT upweights samples in the error set in the following training stage. Besides, LISA also modifies mixup for improving model robustness against distribution shift. However, LISA intuitively mixes the samples within the same subpopulation or same label thus it needs additional subpopulation information. In contrast to them, UMIX introduces sample weights into the vanilla mixup strategy by quantitatively measuring the sample uncertainties without subpopulation information. In addition, our work is orthogonal to LISA, i.e., we can use our weight building strategy to improve LISA's performance. In practice, our method consistently outperforms previous approaches that do not use subpopulation information and even achieves quite competitive performance to those methods which leverage subpopulation information. We also provide theoretical analysis to explain why UMIX works better than the weighted ERM [34, 33, 62, 31].

2 Related work

2.1 Importance weighting

To improve the model robustness against subpopulation shift, importance weighting (IW) is a classical yet effective technique by imposing static or adaptive weight on each sample and then building weighted empirical loss. Therefore each subpopulation group can have a comparable strength in the final training objective. Specifically, there are typically two ways to achieve importance reweighting, i.e., using static or adaptive importance weights.

Static methods. The naive reweighting approaches perform static reweighting based on the distribution of training samples [53, 51, 12, 50, 11, 35]. Their core motivation is to make different subpopulations have a comparable contribution to the training objective by reweighting. Specifically, the most intuitive way is to set the weight of each sample to be inversely proportional to the number of samples in each subpopulation [53, 51, 50]. Besides, there are some methods to obtain sample weights based on the effective number of samples [12], subpopulation margins [11], and Bayesian networks [35].

Adaptive methods. In contrast to the above static methods, a more essential way is to assign each individual sample an adaptive weight that can vary according to training dynamics [57, 64, 39, 62, 28, 40, 33, 54]. Distributional robust optimization (DRO) is one of the most representative methods in this line, which minimizes the loss over the worst-case distribution in a neighborhood of the empirical training distribution. A commonly-used dual form of DRO can be considered as a special case of importance reweighting wherein the sampling weights are updated based on the current loss [44, 19, 31, 20] in an alternated manner. For example, in the group-aware setting (i.e., we know each sample belongs to which subpopulation), GroupDRO [50] introduces an online optimization algorithm to update the weights of each group. In the group-oblivious setting, [57, 28, 39, 40] model the problem as a (regularized) minimax game, where one player aims to minimize the loss by optimizing the model parameters and another player aims to maximize the loss by assigning weights to each sample.

2.2 Uncertainty quantification

The core of our method is based on the high-quality uncertainty quantification of each sample. There are many approaches proposed for this goal. The uncertainty of deep learning models includes epistemic (model) uncertainty and aleatoric (data) uncertainty [24]. To obtain the epistemic uncertainty, Bayesian neural networks (BNNs) [45, 37, 13, 24] is proposed which replace the deterministic weight parameters of model with distribution. Unlike BNNs, ensemble-based methods obtain the epistemic uncertainty by training multiple models and performing ensembles [29, 17, 2, 21]. Aleatoric uncertainty focuses on the inherent noise in the data, which usually is learned as a function of the data [24, 30, 46]. Our method focuses on estimating the uncertainty of training samples with multiple subpopulations and upweighting uncertain samples, thereby improving the performance of minority subpopulations with high uncertainty.

3 Method

In this section, we introduce technical details of UMIX. The key idea of UMIX is to exploit uncertainty information to upweight mixed samples thus can encourage the model to perform uniformly well on all subpopulations. We first introduce the basic procedure of UMIX and then present how to provide high-quality uncertainty estimations which is the fundamental block of UMIX.

3.1 Background

The necessary background and notations are provided here. Let the input and label space be $\mathcal{X}$ and $\mathcal{Y}$ respectively. Given N training samples $\{(x_i, y_i)\}_{i=1}^N$ i.i.d. sampled from a probability distribution P, we consider the setting that there are G predefined subpupulations and the g-th subpopulation follows the distribution P_g. Our goal is to obtain a model $f_\theta : \mathcal{X} \to \mathcal{Y}$ parameterized by $\theta \in \Theta$ that performs well on all subpopulations.

The well-known empirical risk minimization (ERM) algorithm doesn't consider the subpopulations and minimizes the expected risk $\mathbb{E}[\ell(\theta, x_i, y_i)]$, where ℓ denotes the loss function. This leads to the model paying more attention to the majority subpopulations in the training set and resulting in poor performance on the minority subpopulations. For example, the ERM-based models may learn spurious correlations that exist in majority subpopulations but not in minority subpopulations [50]. The proposed method aims to learn a model that is robust against subpopulation shift by importance weighting.

Previous works on improving subpopulation shift robustness investigate several different settings, i.e., group-aware and group-oblivious [62, 34, 50]. Most of the previous works have assumed that the group label is available during training [50, 61]. This is called the group-aware setting. However, due to some reasons (e.g., privacy concerns), we may not have training group labels. This paper studies the group-oblivious setting, which cannot obtain group information for each example at training time. This requires the model to identify underperforming samples and then pay more attention to them during training.

3.2 Importance-weighted mixup

UMIX employs an aggressive data augmentation strategy called uncertainty-aware mixup to mitigate overfitting. Specifically, vanilla mixup [65, 67] constructs virtual training examples (i.e., mixed samples) by performing linear interpolations between data/features and corresponding labels as:

$$\widetilde{x}_{i,j} = \lambda x_i + (1-\lambda)x_j,\ \widetilde{y}_{i,j} = \lambda y_i + (1-\lambda)y_j, \tag{1}$$

where $(x_i, y_i), (x_j, y_j)$ are two samples drawn at random from empirical training distribution and $\lambda \in [0, 1]$. Then vanilla mixup optimizes the following loss function:

$$\mathbb{E}_{\{(x_i,y_i),(x_j,y_j)\}}[\ell(\theta, \widetilde{x}_{i,j}, \widetilde{y}_{i,j})]. \tag{2}$$

When the cross entropy loss is employed, Eq. 2 can be rewritten as:

$$\mathbb{E}_{\{(x_i,y_i),(x_j,y_j)\}}[\lambda\ell(\theta, \widetilde{x}_{i,j}, y_i) + (1-\lambda)\ell(\theta, \widetilde{x}_{i,j}, y_j)]. \tag{3}$$

Eq. 3 can be seen as a linear combination (mixup) of $\ell(\theta, \widetilde{x}_{i,j}, y_i)$ and $\ell(\theta, \widetilde{x}_{i,j}, y_j)$. Unfortunately, since vanilla mixup doesn't consider the subpopulations with poor performance, it has been shown experimentally to be non-robust against subpopulation shift [61]. To this end, we introduce a simple yet effective method called UMIX, which further employs a weighted linear combination of original loss based on Eq. 3 to encourage the learned model to pay more attention to samples with poor performance.

In contrast to previous IW methods, the importance weights of UMIX are posed on the mixed samples. To do this, we first estimate the uncertainty of each sample and then use this quantity to construct the importance weight (i.e., the higher the uncertainty, the higher the weight, and vice versa). For the i-th sample x_i, we denote its importance weight as w_i. Once we obtain the importance weight, we can perform weighted linear combination of $\ell(\theta, \widetilde{x}_{i,j}, y_i)$ and $\ell(\theta, \widetilde{x}_{i,j}, y_j)$ by:

$$\mathbb{E}_{\{(x_i,y_i),(x_j,y_j)\}}[w_i\lambda\ell(\theta, \widetilde{x}_{i,j}, y_i) + w_j(1-\lambda)\ell(\theta, \widetilde{x}_{i,j}, y_j)], \tag{4}$$

where w_i and w_j denote the importance weight of the i-th and j-th samples respectively. In practice, to balance the UMIX and normal training, we set a hyperparameter σ that denotes the probability to apply UMIX. The whole training pseudocode for UMIX is shown in Algorithm 1.

Algorithm 1: The training pseudocode of UMIX.

Input: Training dataset $\mathcal{D}$ and the corresponding importance weights $\mathbf{w} = [w_1, \cdots, w_N]$, hyperparameter σ to control the probability of doing UMIX, and hyperparameter α;

for *each iteration* **do**
- Obtain training samples (x_i, y_i), (x_j, y_j) and the corresponding weight w_i, w_j;
- Sample $p \sim$ Uniform(0,1);
- **if** $p < \sigma$ **then** Sample $\lambda \sim Beta(\alpha, \alpha)$; **else** $\lambda = 0$;
- Obtain the mixed input $\widetilde{x}_{i,j}$ where $\widetilde{x}_{i,j} = \lambda x_i + (1-\lambda)x_j$;
- Obtain the loss of the model with $w_i\lambda\ell(\theta, \widetilde{x}_{i,j}, y_i) + w_j(1-\lambda)\ell(\theta, \widetilde{x}_{i,j}, y_j)$;
- Update model parameters θ to minimize loss with an optimization algorithm.

3.3 Uncertainty-aware importance weights

Now we present how to obtain the uncertainty-aware training importance weights. In the group-oblivious setting, the key to obtaining importance weights is to find samples with high uncertainty. For example, DRO-based algorithms construct the uncertainty set with the current loss [44, 19, 31, 20]. It has been shown experimentally that the uncertain samples found in this way are constantly changing during training [34], resulting in these methods not always upweighting the minority subpopulations. Therefore, we introduce a sampling-based stable uncertainty estimation to better characterize the subpopulation shift.

Given a well trained neural classifier $f_\theta : \mathcal{X} \to \mathcal{Y}$ that could produce the predicted class $\hat{f}_\theta(x)$, a simple way to obtain the uncertainty of a sample is whether the sample is correctly classified. However, as pointed out in previous work [29], a single model cannot accurately characterize the sampling uncertainty. Therefore, we propose to obtain the uncertainty through Bayesian sampling from the model posterior distribution $p(\theta; \mathcal{D})$. Specifically, given a sample (x_i, y_i), we define the training uncertainty as:

$$u_i = \int \kappa(y_i, \hat{f}_\theta(x_i)) p(\theta; \mathcal{D}) d\theta, \text{where } \kappa(y_i, \hat{f}_\theta(x_i)) = \begin{cases} 0, & \text{if } y_i = \hat{f}_\theta(x_i) \\ 1, & \text{if } y_i \neq \hat{f}_\theta(x_i) \end{cases}. \tag{5}$$

Then, we can obtain an approximation of Eq. 5 with T Monte Carlo sampling as $u_i \approx \frac{1}{T} \sum_{t=1}^{T} \kappa(y_i, \hat{f}_{\theta_t}(x_i))$, where $\theta_t \in \Theta$ can be obtained by minimizing the expected risk.

In practice, sampling $\{\theta_t\}_{t=1}^{T}$ from the posterior (i.e., $\theta_t \sim p(\theta; \mathcal{D})$) is computationally expensive and sometimes even intractable since multiple training models need to be built or extra approximation errors need to be introduced. Inspired by a recent Bayesian learning paradigm named SWAG [38], we propose to employ the information from the historical training trajectory to approximate the sampling process. More specifically, we train a model with empirical risk minimization and save the prediction results $\hat{f}_{\theta_t}(x_i)$ of each sample on each iteration epoch t. Then, to avoid the influence of inaccurate predictions at the beginning of training, we estimate uncertainty with predictions after training $T_s - 1$ epochs with:

$$u_i \approx \frac{1}{T} \sum_{t=T_s}^{T_s+T} \kappa(y_i, \hat{f}_{\theta_t}(x_i)). \tag{6}$$

We assume a reasonable importance weight is linearly positively related to the corresponding uncertainty,

$$w_i = \eta u_i + c, \tag{7}$$

where $\eta \in \mathbb{R}_+$ is a hyperparameter and $c \in \mathbb{R}_+$ is a constant that keeps the weight from being 0. In practice, we could set c to 1. The whole process for obtaining training importance weights is shown in Algorithm 2.

Algorithm 2: The process for obtaining training importance weights.

Input: Training dataset $\mathcal{D}$, sampling start epoch T_s, the number of sampling T, and upweight hyperparameter η ;

Output: The training importance weights $\mathbf{w} = [w_1, \cdots, w_n]$;

```
for each iteration do
    Train f_θ by minimizing the expected risk E{ℓ(θ, x_i, y_i)};
    Save the prediction results {f̂_θt(x_i)}_{i=1}^N of the current epoch t;
Obtain the uncertainty of each sample with u_i ≈ (1/T) Σ_{t=T_s}^{T_s+T} κ(y_i, f̂_θt(x_i));
Obtain the importance weight of each sample with w_i = η u_i + c.
```

Rethink why this estimation approach could work? Recent work has empirically shown that compared with the hard-to-classify samples, the easy-to-classify samples are learned earlier during training [15]. Meanwhile, the hard-to-classify samples are also more likely to be forgotten by the neural networks [55]. The frequency with which samples are correctly classified during training can be used as supervision information in confidence calibration [43]. Snapshot performs ensemble learning on several local minima models along the optimization path [21]. The proposed method is also inspired by these observations and algorithms. During training, samples from the minority subpopulations are classified correctly less frequently, which corresponds to higher training uncertainty. On the other hand, samples from the majority subpopulations will have lower training uncertainty due to being classified correctly more often.

4 Experiments

In this section, we conduct experiments on multiple datasets with subpopulation shift to answer the following questions. Q1 Effectiveness (I). In the group-oblivious setting, does the proposed method outperform other algorithms? Q2 Effectiveness (II). Although our method does not use training group labels, does it perform better than the algorithms using training group labels? Q3 Effectiveness (III). Can UMIX improve the model robustness against domain shift where the training and test distributions have different subpopulations. Q4 Qualitative analysis. Are the obtained uncertainties of the training samples trustworthy? Q5 Ablation study. What is the key factor of performance improvement in our method?

4.1 Setup

We briefly present the experimental setup here, including the experimental datasets and comparison methods. Please refer to Sec. B in Appendix for more detailed setup.

Datasets. We perform experiments on three datasets with multiple subpopulations, including Waterbirds [50], CelebA [36] and CivilComments [9]. We also validate UMIX on domain shift scenario which is a more challenging distribution shift problem since there are different subpopulations between training and test data. Hence, we conduct experiments on a medical dataset called Camelyon17 [5, 26] that consists of pathological images from five different hospitals. The training data is drawn from three hospitals, while the validation and test data are sampled from other hospitals.

Evaluation metrics and model selection. To be consistent with existing works [61, 26, 48], we report the average accuracy of Camelyon17 over 10 different seeds. On other datasets, we repeat experiments over 3 times and report the average and worst-case accuracy among all subpopulations. Following prior works [34, 62], we assume the group labels of validation samples are available and select the best model based on worst-case accuracy among all subpopulations on the validation set.

Comparisons in the group-oblivious setting. Here we list the baselines used in the group-oblivious setting. (1) ERM trains the model using standard empirical risk minimization. (2) Focal loss [33] downweights the well-classified examples' loss according to the current classification confidences. (3) DRO-based methods including CVaR-DRO, χ^2-DRO [31], CVaR-DORO and χ^2-DORO [62] minimize the loss over the worst-case distribution in a neighborhood of the empirical training distribution. (4) JTT [34] constructs an error set and upweights the samples in the error set to improve the worst-case performance among all subpopulations.

Comparison in the group-aware setting. To better demonstrate the performance of the proposed method, we compare our method with multiple methods that use training group labels, including IRM [3], IB-IRM [1], V-REx [27], CORAL [32], Group DRO [50], DomainMix [60], Fish [52], and LISA [61].

Mixup-based comparison methods. We compare our method with vanilla mixup and in-group mixup, where vanilla mixup is performed on any pair of samples and in-group mixup is performed on the samples with the same labels and from the same subpopulations.

4.2 Experimental results

We present experimental results and discussions to answer the above-posed questions.

Q1 Effectiveness (I). Since our algorithm does not need training group labels, thus we conduct experiments to verify its superiority over current group-oblivious algorithms. The experimental results are shown in Table 1 and we have the following observations: (1) The proposed UMIX achieves the best worst-case accuracy on all three datasets. For example, for the CelebA dataset, UMIX achieves worst-case accuracy of 85.3%, while the second-best is 81.1%. (2) ERM consistently outperforms other methods in terms of average accuracy. However, it typically comes with the lowest worst-case accuracy. The underlying reason is that the dominance of the majority subpopulations during training leads to poor performance of the minority subpopulations. (3) UMIX shows competitive average accuracy compared to other methods. For example, on CelebA, UMIX achieves the average accuracy of 90.1%, which outperforms all other IW/DRO methods.

Q2 Effectiveness (II). We further conduct comparisons with algorithms that require training group labels. The comparison results are shown in the Table 2. According to the experimental results, it is observed that the performance from our UMIX without using group label is quite competitive compared with these group-aware algorithms. Specifically, benefiting from the uncertainty-aware mixup, UMIX usually performs in the top three in terms of both average and worst-case accuracy.

For example, on WaterBirds, UMIX achieves the best average accuracy of 93.0% and the second-best worst-case accuracy of 90.0%.

Table 1: Comparison results with other methods in the group-oblivious setting. The best results are in bold and blue. Full results with standard deviation are in the Table 6 in Appendix.

	Waterbirds		CelebA		CivilComments		Camelyon17
	Avg.	Worst	Avg.	Worst	Avg.	Worst	Avg.
ERM	**97.0%**	63.7%	**94.9%**	47.8%	**92.2%**	56.0%	70.3%
Focal Loss [33]	87.0%	73.1%	88.4%	72.1%	91.2%	60.1%	68.1%
CVaR-DRO [31]	90.3%	77.2%	86.8%	76.9%	89.1%	62.3%	70.5%
CVaR-DORO [31]	91.5%	77.0%	89.6%	75.6%	90.0%	64.1%	67.3%
χ^2-DRO [62]	88.8%	74.0%	87.7%	78.4%	89.4%	64.2%	68.0%
χ^2-DORO [62]	89.5%	76.0%	87.0%	75.6%	90.1%	63.8%	68.0%
JTT [34]	93.6%	86.0%	88.0%	81.1%	90.7%	67.4%	69.1%
Ours	93.0%	**90.0%**	90.1%	**85.3%**	90.6%	**70.1**%	**75.1%**

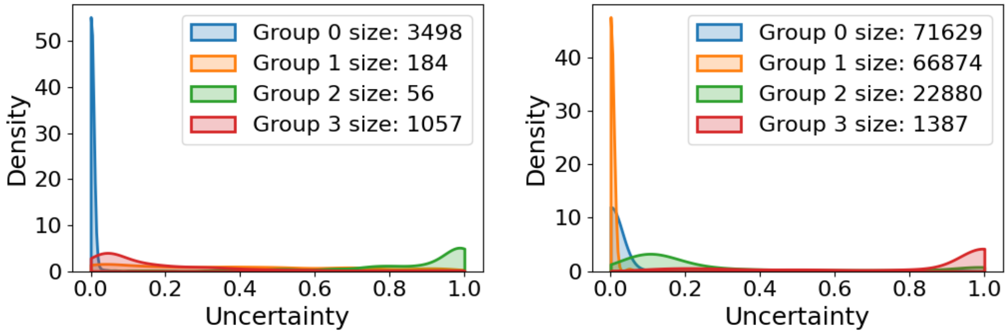
Table 2: Comparison results with the algorithms **using training group labels** (Our method is not dependent on this type of information). Results of baseline models are from [61]. The best three results are in bold brown or bold blue and the color indicates whether the training group labels are used. Full results with standard deviation are in the Table 7 in Appendix.

	Group labels in train set?	Waterbirds		CelebA		CivilComments		Cam17
		Avg.	Worst	Avg.	Worst	Avg.	Worst	Avg.
IRM [3]	Yes	87.5%	75.6%	**94.0%**	77.8%	88.8%	66.3%	64.2%
IB-IRM [1]	Yes	88.5%	76.5%	**93.6%**	85.0%	89.1%	65.3%	68.9%
V-REx [27]	Yes	88.0%	73.6%	92.2%	**86.7%**	**90.2%**	64.9%	71.5%
CORAL [32]	Yes	90.3%	79.8%	**93.8%**	76.9%	88.7%	65.6%	59.5%
GroupDRO [50]	Yes	**91.8%**	**90.6%**	92.1%	**87.2%**	89.9%	70.0%	68.4%
DomainMix [60]	Yes	76.4%	53.0%	93.4%	65.6%	**90.9%**	63.6%	69.7%
Fish [52]	Yes	85.6%	64.0%	93.1%	61.2%	89.8%	**71.1%**	**74.7%**
LISA [61]	Yes	91.8%	**89.2%**	92.4%	**89.3%**	89.2%	**72.6%**	**77.1%**
Ours	**No**	**93.0%**	**90.0%**	90.1%	85.3%	**90.6%**	**70.1%**	**75.1%**

Q3 Effectiveness (III). We conduct comparison experiments on Camelyon17 to investigate the effectiveness of our algorithm under the domain shift scenario. The experimental results are shown in the last column of Table 1 and Table 2 respectively. In the group-oblivious setting, the proposed method achieves the best average accuracy on Camelyon17 as shown in Table 1. For example, UMIX achieves the best average accuracy of 75.1% while the second is 70.3%. Meanwhile, in Table 2, benefiting from upweighting the mixed samples with poor performance, our method achieves a quite competitive generalization ability on Camelyon17 compared with other algorithms using training group labels.

Q4 Qualitative analysis. To intuitively investigate the rationality of the estimated uncertainty, we visualize the density of the uncertainty for different groups with kernel density estimation. As shown in Fig. 1, the statistics of estimated uncertainty is basically correlated to the training sample size of each group. For example, on Waterbirds and CelebA, the average uncertainties of minority groups are much higher, while those of majority groups are much lower.

Q5 Ablation study. Finally, we conduct the ablation study in comparison with vanilla mixup and in-group mixup. The experimental results are shown in Table 3. Compared with ERM, vanilla mixup cannot significantly improve worst-case accuracy. After using the group label, the in-group mixup slightly improves the worst-case accuracy compared to ERM. The possible reason is that mixup-based methods do not increase the influence of minority subpopulations in the model objective function. Although our method does not use the group label of the training samples, our method still can significantly improve the worst-case accuracy.



(a) Waterbirds (b) CelebA

Figure 1: Visualization of the obtained uncertainty with kernel density estimation on Waterbirds and CelebA datasets, where group size refers to the sample number of the group.

Table 3: Comparison with ERM and mixup based methods. Results of baseline models are from [61]. The best results are in bold brown or bold blue and the color indicates whether the training group labels are used. Full results with standard deviation are in the Table 8 in Appendix.

	Group labels in train set?	Waterbirds Avg.	Waterbirds Worst	CelebA Avg.	CelebA Worst	CivilComments Avg.	CivilComments Worst	Cam17 Avg.
ERM	No	**97.0%**	63.7%	94.9%	47.8%	**92.2%**	56.0%	70.3%
vanilla mixup	No	81.0%	56.2%	**95.8%**	46.4%	90.8%	67.2%	71.2%
in-group mixup	Yes	88.7%	68.0%	95.2%	58.3%	90.8%	69.2%	**75.5%**
Ours	No	93.0%	**90.0%**	90.1%	**85.3%**	90.6%	**70.1%**	75.1%

5 Theory

In this section, we provide a theoretical understanding of the generalization ability for UMIX. At a high level, we prove that our method can achieve a better generalization error bound than traditional IW methods without using mixup. For simplicity, our analysis focuses on generalized linear model (GLM). The roadmap of our analysis is to first approximate the mixup loss and then study the generalization bound from a Rademacher complexity perspective. To introduce the theoretical framework, we first present the basic settings.

Basic settings. Our analysis mainly focuses on GLM model classes whose loss function ℓ follows $\ell(\theta, x, y) = A(\theta^\top x) - y\theta^\top x$, where $x \in \mathbb{R}^d$ is the input , $\theta \in \mathbb{R}^d$ is the parameter, $y \in \mathbb{R}$ is the label and $A(\cdot)$ is the log-partition function. We impose the following assumptions on $A(\cdot)$ which can be widely satisfied by the most commonly used GLMs, e.g., logistical regression and linear model.

Assumption 5.1. *We assume $A(\cdot)$ is twice differentiable and for all $|z| \leq 1$, there exists some $K > 0$ such that $K^{-1} \leq A''(z) \leq K$. Moreover, we assume $\|\theta\| \leq 1$.*

Recall the setting of subpopulation shift, we assume that the population distribution P consists of G different subpopulations with the g-th subpopulation's proportion is k_g and the g-th subpopulation follows the distribution P_g. In specific, we have $P = \sum_{g=1}^{G} k_g P_g$. Then we denote the covariance matrix for the g-th subpopulation as $\Sigma_X^g = \mathbb{E}_{(x,y)\sim P_g}[xx^T]$. For simplicity, we consider the case where a shared weight w_g is assigned to all samples from the g-th subpopulation. The main goal of our theoretical analysis is to characterize the generalization ability of the model learned using Algorithm 1. Formally, we focus on analyzing the upper bound of the weighted generalization error defined as:

$$\mathrm{GError}(\theta) = \mathbb{E}_{(x,y)\sim P}[w(x,y)\ell(\theta,x,y)] - \frac{1}{N}\sum_{i=1}^{N} w(x_i,y_i)\ell(\theta,x_i,y_i),$$

where the function $w(x, y)$ is the weighted function to return the weight of the subpopulation to which the sample (x, y) belongs.

First of all, we present our main result in this section. The main theorem of our analysis provides a subpopulation-heterogeneity dependent bound for the above generalization error. This theorem is formally presented as:

Theorem 5.1. *Suppose $A(\cdot)$ is L_A-Lipschitz continuous, then there exists constants $L, B > 0$ such that for any θ satisfying $\theta^\top \Sigma_X \theta \leq \gamma$, the following holds with a probability of at least $1 - \delta$,*

$$\text{GError}(\theta) \leq 2L \cdot L_A \cdot (\max\{(\frac{\gamma(\delta/2)}{\rho})^{1/4}, (\frac{\gamma(\delta/2)}{\rho})^{1/2}\} \cdot \sqrt{\frac{\text{rank}(\Sigma_X)}{n}}) + B\sqrt{\frac{\log(2/\delta)}{2n}},$$

where $\gamma(\delta)$ is a constant dependent on δ and $\Sigma_X = \sum_{g=1}^{G} k_g w_g \Sigma_X^g$.

We will show later that the output of our Algorithm 1 can satisfy the constraint $\theta^\top \Sigma_X \theta \leq \gamma$ and thus Theorem 5.1 can provide a theoretical understanding of our algorithm. In contrast to weighted ERM, the bound improvement of UMIX is on the red term which can partially reflect the heterogeneity of the training subpopulations. Specifically, the red term would become $\sqrt{d/n}$ in the weighted ERM setting (see more detailed theoretical comparisons in Appendix.) Thus our bound can be tighter when the intrinsic dimension of data is small (i.e., $\text{rank}(\Sigma) \ll d$).

The proof of Theorem 5.1 follows this roadmap: (1) We first show that the model learned with UMIX can fall into a specific hypothesis set $\mathcal{W}_\gamma$. (2) We analyze the Rademacher complexity of the hypothesis set and obtain its complexity upper bound (Lemma A.3). (3) Finally, we can characterize the generalization bound by using complexity-based learning theory [7] (Theorem 8). More details of the proof can be found in Appendix.

As we discuss in Appendix, the weighted mixup can be seen as an approximation of a regularization term $\frac{C}{n}[\sum_{i=1}^{n} w_i A''(x_i^\top \theta)]\theta^\top \widehat{\Sigma}_X \theta$ for some constant C compared with the non-mixup algorithm, which motivates us to study the following hypothesis space

$$\mathcal{W}_\gamma := \{x \to \theta^\top x, \text{such that } \theta \text{ satisfying } \mathbb{E}_{x,y}[w(x, y)A''(x^\top \theta)]\theta^\top \Sigma_X \theta \leq \gamma\},$$

for some constant γ.

To further derive the generalization bound, we also need the following assumption, which is satisfied by general GLMs when θ has bounded ℓ_2 norm and it is adopted in, e.g., [4, 67].

Assumption 5.2 (ρ-retentive)**.** *We say the distribution of x is ρ-retentive for some $\rho \in (0, 1/2]$ if for any non-zero vector $v \in \mathbb{R}^d$ and given the event that $\theta \in \mathcal{W}_\gamma$ where the θ is output by our Algorithm 1, we have*

$$\mathbb{E}_x^2[A''(x^\top v)] \geq \rho \cdot \min\{1, \mathbb{E}_x(v^\top x)^2\}.$$

Finally, we can derive the Rademacher complexity of the $\mathcal{W}_\gamma$ and the proof of Theorem 5.1 is obtained by combining Lemma A.3 and the Theorem 8 of [7].

Lemma 5.1. *Assume that the distribution of x_i is ρ-retentive, i.e., satisfies the assumption 5.2. Then the empirical Rademacher complexity of $\mathcal{W}_r$ satisfies*

$$Rad(\mathcal{W}_r, \mathcal{S}) \leq \max\{(\frac{\gamma(\delta)}{\rho})^{1/4}, (\frac{\gamma(\delta)}{\rho})^{1/2}\} \cdot \sqrt{\frac{rank(\Sigma_X)}{n}},$$

with probability at least $1 - \delta$.

6 Conclusion

In this paper, we propose a novel method called UMIX to improve the model robustness against subpopulation shift. We propose a simple yet reliable approach to estimate the sample uncertainties and integrate them into the mixup strategy so that UMIX can mitigate the overfitting thus improving prior IW methods. Our method consistently outperforms previous approaches on commonly-used benchmarks. Furthermore, UMIX also shows the theoretical advantage that the learned model comes with subpopulation-heterogeneity dependent generalization bound. In the future, how to leverage subpopulation information to improve UMIX can be a promising research direction.

References

[1] Kartik Ahuja, Ethan Caballero, Dinghuai Zhang, Jean-Christophe Gagnon-Audet, Yoshua Bengio, Ioannis Mitliagkas, and Irina Rish. Invariance principle meets information bottleneck for out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 34, 2021.

[2] Javier Antorán, James Urquhart Allingham, and José Miguel Hernández-Lobato. Depth uncertainty in neural networks. In *Advances in Neural Information Processing Systems*, 2020.

[3] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

[4] Raman Arora, Peter Bartlett, Poorya Mianjy, and Nathan Srebro. Dropout: Explicit forms and capacity control. In *International Conference on Machine Learning*, pages 351–361. PMLR, 2021.

[5] Peter Bandi, Oscar Geessink, Quirine Manson, Marcory Van Dijk, Maschenka Balkenhol, Meyke Hermsen, Babak Ehteshami Bejnordi, Byungjae Lee, Kyunghyun Paeng, Aoxiao Zhong, et al. From detection of individual metastases to classification of lymph node status at the patient level: the camelyon17 challenge. *IEEE transactions on medical imaging*, 38(2):550–560, 2018.

[6] Solon Barocas and Andrew D Selbst. Big data's disparate impact. *Calif. L. Rev.*, 104:671, 2016.

[7] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[8] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of the 24th international conference on Machine learning*, pages 81–88, 2007.

[9] Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion proceedings of the 2019 world wide web conference*, pages 491–500, 2019.

[10] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *International Conference on Machine Learning*, pages 872–881. PMLR, 2019.

[11] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32, 2019.

[12] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9268–9277, 2019.

[13] John Denker and Yann LeCun. Transforming neural-net output levels to probability distributions. *Advances in neural information processing systems*, 3, 1990.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

[15] Yonatan Geifman, Guy Uziel, and Ran El-Yaniv. Bias-reduced uncertainty estimation for deep neural classifiers. In *International Conference on Learning Representations*, 2019.

[16] Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pages 1929–1938. PMLR, 2018.

[17] Marton Havasi, Rodolphe Jenatton, Stanislav Fort, Jeremiah Zhe Liu, Jasper Snoek, Balaji Lakshminarayanan, Andrew Mingbo Dai, and Dustin Tran. Training independent subnetworks for robust prediction. In *International Conference on Learning Representations*, 2021.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[19] Weihua Hu, Gang Niu, Issei Sato, and Masashi Sugiyama. Does distributionally robust supervised learning give robust classifiers? In *International Conference on Machine Learning*, pages 2029–2037. PMLR, 2018.

[20] Zhaolin Hu and L Jeff Hong. Kullback-leibler divergence constrained distributionally robust optimization. *Available at Optimization Online*, pages 1695–1724, 2013.

[21] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft, and Kilian Q Weinberger. Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*, 2017.

[22] Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex Smola. Correcting sample selection bias by unlabeled data. *Advances in neural information processing systems*, 19, 2006.

[23] Nathalie Japkowicz. The class imbalance problem: Significance and strategies. In *Proc. of the Int'l Conf. on Artificial Intelligence*, volume 56, pages 111–117. Citeseer, 2000.

[24] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

[25] Allison Koenecke, Andrew Nam, Emily Lake, Joe Nudell, Minnie Quartey, Zion Mengesha, Connor Toups, John R Rickford, Dan Jurafsky, and Sharad Goel. Racial disparities in automated speech recognition. *Proceedings of the National Academy of Sciences*, 117(14):7684–7689, 2020.

[26] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

[27] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). In *International Conference on Machine Learning*, pages 5815–5826. PMLR, 2021.

[28] Preethi Lahoti, Alex Beutel, Jilin Chen, Kang Lee, Flavien Prost, Nithum Thain, Xuezhi Wang, and Ed Chi. Fairness without demographics through adversarially reweighted learning. *Advances in neural information processing systems*, 33:728–740, 2020.

[29] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017.

[30] Quoc V Le, Alex J Smola, and Stéphane Canu. Heteroscedastic gaussian process regression. In *Proceedings of the 22nd international conference on Machine learning*, pages 489–496, 2005.

[31] Daniel Levy, Yair Carmon, John C Duchi, and Aaron Sidford. Large-scale methods for distributionally robust optimization. *Advances in Neural Information Processing Systems*, 33:8847–8860, 2020.

[32] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5400–5409, 2018.

[33] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[34] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pages 6781–6792. PMLR, 2021.

[35] Wei Liu and Sanjay Chawla. Class confidence weighted knn algorithms for imbalanced data sets. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 345–356. Springer, 2011.

[36] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

[37] David JC MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

[38] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[39] Paul Michel, Tatsunori Hashimoto, and Graham Neubig. Modeling the second player in distributionally robust optimization. In *ICLR 2021*, 2021.

[40] Paul Michel, Tatsunori Hashimoto, and Graham Neubig. Distributionally robust models with parametric likelihood ratios. In *International Conference on Learning Representations*, 2022.

[41] Microsoft. Neural Network Intelligence, 1 2021.

[42] John Stuart Mill. Utilitarianism. In *Seven masterpieces of philosophy*, pages 337–383. Routledge, 2016.

[43] Jooyoung Moon, Jihyo Kim, Younghak Shin, and Sangheum Hwang. Confidence-aware learning for deep neural networks. In *international conference on machine learning*, pages 7034–7044. PMLR, 2020.

[44] Hongseok Namkoong and John C Duchi. Stochastic gradient methods for distributionally robust optimization with f-divergences. *Advances in neural information processing systems*, 29, 2016.

[45] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[46] David A Nix and Andreas S Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 ieee international conference on neural networks (ICNN'94)*, volume 1, pages 55–60. IEEE, 1994.

[47] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[48] Vihari Piratla, Praneeth Netrapalli, and Sunita Sarawagi. Focus on the common good: Group distributional robustness follows. In *International Conference on Learning Representations*, 2022.

[49] John Rawls. *Justice as fairness: A restatement*. Harvard University Press, 2001.

[50] Shiori Sagawa*, Pang Wei Koh*, Tatsunori B. Hashimoto, and Percy Liang. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2020.

[51] Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. An investigation of why overparameterization exacerbates spurious correlations. In *International Conference on Machine Learning*, pages 8346–8356. PMLR, 2020.

[52] Yuge Shi, Jeffrey Seely, Philip HS Torr, N Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. *arXiv preprint arXiv:2104.09937*, 2021.

[53] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

[54] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *Advances in neural information processing systems*, 32, 2019.

[55] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019.

[56] Joel A Tropp. An introduction to matrix concentration inequalities. *arXiv preprint arXiv:1501.01571*, 2015.

[57] Junfeng Wen, Chun-Nam Yu, and Russell Greiner. Robust learning under uncertain test distributions: Relating covariate shift to model misspecification. In *International Conference on Machine Learning*, pages 631–639. PMLR, 2014.

[58] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[59] Da Xu, Yuting Ye, and Chuanwei Ruan. Understanding the role of importance weighting for deep learning. In *International Conference on Learning Representations*, 2021.

[60] Minghao Xu, Jian Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. Adversarial domain adaptation with domain mixup. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6502–6509, 2020.

[61] Huaxiu Yao, Yu Wang, Sai Li, Linjun Zhang, Weixin Liang, James Zou, and Chelsea Finn. Improving out-of-distribution robustness via selective augmentation. In *Proceeding of the Thirty-ninth International Conference on Machine Learning*, 2022.

[62] Runtian Zhai, Chen Dan, Zico Kolter, and Pradeep Ravikumar. Doro: Distributional and outlier robust optimization. In *International Conference on Machine Learning*, pages 12345–12355. PMLR, 2021.

[63] Runtian Zhai, Chen Dan, Zico Kolter, and Pradeep Ravikumar. Understanding why generalized reweighting does not improve over erm. *arXiv preprint arXiv:2201.12293*, 2022.

[64] Runtian Zhai, Chen Dan, Arun Suggala, J Zico Kolter, and Pradeep Ravikumar. Boosted cvar classification. *Advances in Neural Information Processing Systems*, 34, 2021.

[65] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[66] Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. How does mixup help with robustness and generalization? *arXiv preprint arXiv:2010.04819*, 2020.

[67] Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. How does mixup help with robustness and generalization? In *International Conference on Learning Representations*, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Sec. C in Appendix.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Sec. C in Appendix.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Sec. 5.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Sec. A in Appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Code will be released after the paper is accepted.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Sec. B in Appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Sec. B in Appendix.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Sec. B in Appendix.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] The datasets used are all publicly available datasets.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] The datasets used are all publicly available datasets.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] We didn't conduct research with human subjects.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] We didn't conduct research with human subjects.
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] We didn't conduct research with human subjects.