Minority Oversampling for Imbalanced Data via Class-Preserving Regularized Auto-Encoders

Arnab Kumar Mondal
IIT Delhi

Lakshya Singhal
IISc Bengaluru

Piyush Tiwary
IISc Bengaluru

Parag Singla
IIT Delhi

Prathosh AP
IISc Bengaluru

Abstract

Class imbalance is a common phenomenon in multiple application domains such as healthcare, where the sample occurrence of one or few class categories is more prevalent in the dataset than the rest. This work addresses the class-imbalance issue by proposing an over-sampling method for the minority classes in the latent space of a Regularized Auto-Encoder (RAE). Specifically, we construct a latent space by maximizing the conditional data likelihood using an Encoder-Decoder structure, such that oversampling through convex combinations of latent samples preserves the class identity. A jointly-trained linear classifier that separates convexly coupled latent vectors from different classes is used to impose this property on the AE's latent space. Further, the aforesaid linear classifier is used for final classification without retraining. We theoretically show that our method can achieve a low variance risk estimate compared to naive oversampling methods and is robust to overfitting. We conduct several experiments on benchmark datasets and show that our method outperforms the existing oversampling techniques for handling class imbalance. The code of the proposed method is available at: `https://github.com/arnabkmondal/OversamplingRAE`

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206. Copyright 2023 by the author(s).

1 Introduction

1.1 Background and Motivation

The framework of empirical risk minimization (Shalev-Shwartz & Ben-David, 2014) for classification does not yield good estimates in the presence of class imbalance, which refers to having a skewed distribution over the class or label priors (Wei *et al.*, 2013; Yuan *et al.*, 2018; Kubat *et al.*, 1998; Rao *et al.*, 2006). Many real-world applications such as fraud/fault detection (Wei *et al.*, 2013) and clinical anomaly detection (Yuan *et al.*, 2018) inherently possess the problem of imbalance because the class of interest will be naturally sparse. In such cases, the usually employed monte-carlo estimates for class conditional empirical risk for the minority classes will be very biased, leading to overfitting on minority class samples (Li *et al.*, 2020). Therefore, it is important to address the problem of class imbalance in real-world classification problems appearing in application areas such as healthcare.

The problem of class-imbalance is well recognised in the literature and several solutions have been proposed (see Sec. 2 for a detailed discussion). Two of the most common approaches involve loss function modification (Li *et al.*, 2021; Cao *et al.*, 2019; Wang *et al.*, 2016; Lin *et al.*, 2017a; Zhang *et al.*, 2017b; Sarafianos *et al.*, 2018; Dong *et al.*, 2018; Li *et al.*, 2019; Cui *et al.*, 2019; Park *et al.*, 2021; Kini *et al.*, 2021), and data resampling (Hart, 1968; Wilson, 1972; Tomek, 1976; Laurikkala, 2001; Mani & Zhang, 2003; Kubat & Matwin, 1997; García & Herrera, 2009; Koziarski, 2020; Lin *et al.*, 2017b; Vuttipittayamongkol & Elyan, 2020; Chawla *et al.*, 2002; Han *et al.*, 2005; Nguyen *et al.*, 2011; He *et al.*, 2008; Stefanowski & Wilk, 2008; Abdi & Hashemi, 2016; Batista *et al.*, 2004; Ramentol *et al.*, 2012; Sáez *et al.*, 2015; Mariani *et al.*, 2018; Mullick *et al.*, 2019; Wang *et al.*, 2020; Dablain *et al.*, 2022). In the former class of methods, the objective is to design novel loss functions that suit better for long tailed distributions whereas in latter methods, the objective is to balance the dataset via resampling. The current work focuses on proposing a new resampling (specifically oversampling) strategy *that could address the shortcomings of the existing resampling methods*.

The objective of the oversampling methods is to learn the distribution of the minority classes and oversample from it to reduce the inherent skewness present in the dataset. Classical methods such as *synthetic minority oversampling technique* (SMOTE) (Chawla *et al.*, 2002) algorithm synthetically generates data from the minority class by interpolations between the nearest neighbors of a given point from the minority class. Despite being intuitive and simple, they are not very effective with high dimensional data (e.g., images) owing to the 'curse of dimensionality' (Dablain *et al.*, 2022). Recently, deep generative models such as Generative Adversarial Networks (GANs) (Goodfellow *et al.*, 2014) and Variational Auto-Encoders (Kingma & Welling, 2013) have been used to address the class imbalance problem for high dimensional data by conditionally generating more samples from the minority class. While they improve upon the classical methods by a considerable margin, it may result in a lack of diversity in the generated minority samples (due to mode collapse (Shahbazi *et al.*, 2022)), especially in the limited data regime. Additionally, retraining on the oversampled points may or may not aid the final classification since, in many of such methods (Dablain *et al.*, 2022; Mariani *et al.*, 2018), the classifier and sampler operate independently.

Motivated by the above observations, we seek to build a 'good' oversampling method with the following properties:

- Oversampling is carried out in a lower-dimensional latent space than in the original data space (unlike (Chawla *et al.*, 2002; Han *et al.*, 2005; Mullick *et al.*, 2019)) to avoid the 'curse of dimensionality'.
- The oversampling strategy is independent of an explicit distance metric and is class preserving by construction (unlike in SMOTE (Chawla *et al.*, 2002) and DeepSMOTE (Dablain *et al.*, 2022)). In other words, the latent space is learned in a way that is conducive to oversampling and classification.
- Instead of learning the decision boundary (of the classifier) after minority oversampling (like BAGAN (Mariani *et al.*, 2018) and DeepSMOTE (Dablain *et al.*, 2022)), oversampling should be done simultaneously while learning the decision boundary. We believe this will lead to a robust classifier.

1.2 Contributions

Our goal is to learn a low-dimensional representation of data that facilitates oversampling in a class-preserving manner. Specifically, the contributions of this work are listed as follows:

1. We present a method for oversampling that uses convex combinations of the latent vectors of a regularized autoencoder learned by maximizing conditional data likelihood.
2. Under the proposed oversampling technique, the latent space is regularized to preserve the class (via construction) by concurrently learning a linear classifier on the latent space along with the mixing coefficients required for oversampling.
3. We theoretically establish that the class preserving minority oversampling via convex combinations reduces the variance of the estimated empirical risk and results in a robust classifier in terms of the Lipschitz constant.
4. We conduct extensive experiments on several datasets showing superior performance over many recent benchmark methods to validate our claims. We observe a gain in ACSA of about **12%**, **8%** and **7%** on CIFAR10, ImageNet-100 and CelebA datasets respectively.

2 Related Work

The approaches for mitigating class-imbalance can broadly be classified into following categories: data resampling methods (Hart, 1968; Wilson, 1972; Tomek, 1976; Laurikkala, 2001; Mani & Zhang, 2003; Kubat & Matwin, 1997; García & Herrera, 2009; Koziarski, 2020; Lin *et al.*, 2017b; Vuttipittayamongkol & Elyan, 2020; Chawla *et al.*, 2002; Han *et al.*, 2005; Nguyen *et al.*, 2011; He *et al.*, 2008; Stefanowski & Wilk, 2008; Abdi & Hashemi, 2016; Batista *et al.*, 2004; Ramentol *et al.*, 2012; Sáez *et al.*, 2015), feature resampling methods (Ando & Huang, 2017; Chu *et al.*, 2020), learning-objective modification methods (Li *et al.*, 2021; Cao *et al.*, 2019; Wang *et al.*, 2016; Lin *et al.*, 2017a; Zhang *et al.*, 2017b; Sarafianos *et al.*, 2018; Dong *et al.*, 2018; Li *et al.*, 2019; Cui *et al.*, 2019; Park *et al.*, 2021; Kini *et al.*, 2021), cost-sensitive learning methods (Domingos, 1999; Fan *et al.*, 1999; Karakoulas & Shawe-Taylor, 1998; Viola & Jones, 2002; Ting, 2002; Zhou & Liu, 2006), Meta-learning approaches (Wang *et al.*, 2017; Wu *et al.*, 2018; Shu *et al.*, 2019; Lee *et al.*, 2020; Liu *et al.*, 2020a), ensemble learning (Liu *et al.*, 2008, 2020b; Freund & Schapire, 1997; Guo & Viktor, 2004; Chawla *et al.*, 2003; Seiffert *et al.*, 2009; Wang & Yao, 2012; Galar *et al.*, 2013; Barandela *et al.*, 2003; Wang & Yao, 2009; Błaszczyński *et al.*, 2010), semi and self-supervised learning approaches, (Yang & Xu, 2020; Kim *et al.*, 2020; Lee *et al.*, 2021), curriculum learning based methods (Hu *et al.*, 2019; Zheng *et al.*, 2018; Wang *et al.*, 2019), data-augmentation methods (Zhao & Lei, 2021) and representation learning based approaches (Huang *et al.*, 2016; Kang *et al.*, 2020; Chen *et al.*, 2021). Here, we review only the relevant data resampling literature in detail. For an extensive review of other methods refer to the latest survey by Das *et al.* (2022).

Classical approaches for oversampling: These resampling methods are classifier-independent in that they first re-sample without involving the classifier, which is trained with the balanced data post hoc. They can be broadly divided into three categories: Undersampling methods (Hart, 1968; Wilson, 1972; Tomek, 1976; Laurikkala, 2001; Mani & Zhang, 2003; Kubat & Matwin, 1997; García & Herrera, 2009; Koziarski, 2020; Lin *et al.*, 2017b; Vuttipittayamongkol & Elyan, 2020) that reduce the size of majority class, Oversampling methods (Chawla *et al.*, 2002; Han *et al.*, 2005; Nguyen *et al.*, 2011; He *et al.*, 2008; Stefanowski & Wilk, 2008; Abdi & Hashemi, 2016) that increase the size of minority class, and Hybrid methods (Batista *et al.*, 2004; Ramentol *et al.*, 2012; Sáez *et al.*, 2015) that combine both under and over sampling. Out of these three strategies, oversampling has gained a lot of attention due to the remarkable success achieved by Synthetic Minority Over-sampling TEchnique (SMOTE) (Chawla *et al.*, 2002) and ADAptive SYNthetic (ADASYN) sampling (He *et al.*, 2008). Both of these techniques generate novel data instances from the minority class by interpolation between a given samples and its nearest neighbors defined by a certain distance metric. ADASYN seeks to generate examples close to the original samples that were incorrectly classified using a k-Nearest Neighbors classifier. Whereas, the standard SMOTE does not distinguish between easy and hard samples to be classified. A few variants of SMOTE (Han *et al.*, 2005; Nguyen *et al.*, 2011) focus on samples near the optimal decision boundary and generate samples in the opposite direction of the nearest neighbors' class. However, for high-dimensional data such as images, these classical pixel-based oversampling methods not only suffer from 'curse of dimensionality' but also are memory intensive as they require access to the entire training split for computing nearest neighbors. Recent research (Koziarski *et al.*, 2019) has shown that SMOTE-based algorithms cannot handle multi-modal data with a large intra-class overlap or noise. To mitigate this issue, Radial-Based Oversampling (RBO) (Koziarski *et al.*, 2019) avoids using k-nearest neighbors in favour of imbalance distribution estimation with radial basis functions.

Deep Generative Model-based Resampling approaches: Deep neural generative frameworks such as GAN (Goodfellow *et al.*, 2014), VAE (Kingma & Welling, 2013) and their variants (Arjovsky *et al.*, 2017; Gulrajani *et al.*, 2017; Makhzani *et al.*, 2016; Tolstikhin *et al.*, 2018; Dai & Wipf, 2019; Mondal *et al.*, 2020, 2021) have been extensively utilized for sampling from high dimensional data (Yi *et al.*, 2019). Owing to this, the work in (Douzas & Bacao, 2018) uses conditional Generative Adversarial Networks (cGAN) (Mirza & Osindero, 2014) to generate data from the minority class of various imbalanced datasets. However, such a framework may not be ideal (Mariani *et al.*, 2018; Mullick *et al.*, 2019) due to boundary distortion (Santurkar *et al.*, 2018) and mode-collapse (Shahbazi *et al.*, 2022). To overcome these problems, BAGAN (Mariani *et al.*, 2018) uses an autoencoder to initialize the GAN modules and includes all accessible images of majority and minority classes during adversarial training; this allows the generative model to acquire valuable features from majority classes and apply them to minority class images. Generative Adversarial Minority Oversampling (GAMO) (Mullick *et al.*, 2019) exploits a three-player adversarial game among a convex generator, a classifier, and a discriminator. The convex generator creates new samples from minority classes as convex combinations of existing examples (in the pixel space) to deceive both the discriminator and the classifier into misidentifying the generated samples. As a result, synthetic data is generated at critical locations around the peripheries of different classes. Consequently, the classifier-induced boundaries are adjusted in such a way that the minority classes are less likely to be miscategorized. Deep Generative Classifier (DGC) (Wang *et al.*, 2020) is a deep latent variable model that aims to capture the cause of the target label in the latent variable. To mitigate the class imbalance problem, DGC takes advantage of both model perturbation and data perturbation. DeepSMOTE (Dablain *et al.*, 2022) uses the reconstruction loss and a supervised penalty to train an encoder-decoder pair and then performs SMOTE in the latent space during inference.

Uniqueness of our work: The uniqueness of our work lies in the formulation and realization of "class preserving oversampling". In particular, we layout the theoretical framework to achieve class preserving oversampling (Section 3.1) and then we show that the proposed formulation can be realized using a Regularized Auto Encoder (RAE) (Section 3.2). To ensure class preservation, we regularise the latent space of an auto-encoder using convex combinations and a linear classifier. The concept of employing a linear classifier to regularise the latent space of an auto-encoder is not new; for instance, (Le *et al.*, 2018) uses a linear projection of the latent vector to solve a regression task along with reconstructing the input to obtain high performance on the regression task. However, there is no concept of class-preservation involved in this work. Similarly, using convex combination of latent vectors has also been explored in prior literature. (Zhang *et al.*, 2017a; Verma *et al.*, 2019) show that using a convex combination of feature maps for training helps in obtaining meaningful representations which is further helpful in downstream tasks. (Berthelot *et al.*, 2018) uses convex combination of latent vector to generate a real sample from the decoder, they show that this aids in learning meaningful representations and it makes the latent space 'smooth'. However, none of these works ensure class preservation. Further-more, we note that while the RAE described in this work is one way of realising the suggested formulation, there may be additional methods as well.

3 Proposed Method

3.1 Class Preserving Oversampling

Our goal in this work is to learn latent representations of the input data such that we can sample from it in a class-preserving manner. Mathematically, consider a labelled dataset $\mathcal{D}_{\mathcal{X}\times\mathcal{Y}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N} \overset{iid}{\sim} \boldsymbol{p}(x, y)$, where, $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{1, \cdots C\}$ denotes i^{th}-datapoint and its corresponding class-label, respectively with $\boldsymbol{p}(x, y)$ denoting their true joint distribution. Given $\mathcal{D}_{\mathcal{X}\times\mathcal{Y}}$, our objective is to learn a latent space $\mathcal{Z} \in \mathbb{R}^m$ with a corresponding distribution $\boldsymbol{q}(\mathbf{z})$ such that oversampling in the $\mathcal{Z}$ space preserves the class labels.

We propose to oversample in the $\mathcal{Z}$ space by taking convex combinations of latent vectors corresponding to the same class. Let $\mathbf{z}_i, i = 1, 2, ..., t$ denote the latent vectors corresponding to datapoints $\mathbf{x}_i, i = 1, 2, ..., t$ all with the same class-label. Then the novel (oversampled) latent point $\mathbf{z}'$ is obtained as follows:

$$\mathbf{z}' = \sum_{i=1}^{t} \alpha_i \mathbf{z}_i \qquad \text{s.t.} \quad \sum_i \alpha_i = 1 \quad \text{and} \quad \alpha_i \geq 0 \quad (1)$$

However, such a method for sampling does not enforce class preservation, that is, the class label of $\mathbf{z}'$ may not be same as that of $\mathbf{z}_i$. Classes of the oversampled latents can be preserved by learning a latent space such that the class-conditional latent densities have non-overlapping supports and the classes are linearly separable in the latent space. Formally, let $R_i = \{\mathbf{x} \mid h(\mathbf{z}) = i\}$ denote the set of all input samples $\mathbf{x}$ that gets mapped to class $y = i$ under an oracle linear classifier $h(\mathbf{z})$. Suppose $\boldsymbol{q}(\mathbf{z} \mid \mathbf{x} \in R_i)$ represent the class-conditional latent distribution for class i, then we seek $Supp\big(\boldsymbol{q}(\mathbf{z} \mid \mathbf{x} \in R_i)\big) \cap Supp\big(\boldsymbol{q}(\mathbf{z} \mid \mathbf{x} \in R_j)\big) = \phi$ $\forall\, i \neq j$ where, $Supp(\cdot)$ denotes the support of the distribution. Practically, this can be achieved by jointly learning the latent space along with a *linear classifier* that assigns the same label to latent vectors corresponding to a given class and their convexly combined (oversampled) counterparts (Proposition 1). We propose to achieve it in a regularized auto encoder framework as described in the next section.

3.2 Regularized Autoencoders with class preserving latent space

The aforementioned objective can be potentially achieved by inducing a latent space where all the data points corresponding to a particular class are mapped to a single latent vector. This results in producing de-generate representations, which defeats the purpose of oversampling. To avoid such de-generate representations, we propose to learn the latent space by maximizing the conditional data-likelihood $\boldsymbol{p}(\boldsymbol{x} \mid \boldsymbol{z}')$ such that the class-preservation constraints are met. Specifically, we parameterize the data distribution conditioned on the oversampled latent space with a decoder network $\boldsymbol{p}_\theta(\mathbf{x} \mid \mathbf{z}')$ and the conditioned latent distribution using an encoder network $\boldsymbol{q}_\varphi(\mathbf{z} \mid \mathbf{x})$. Additionally, the class-preserving constraint detailed in the previous section is imposed on the latent space (output of the encoder) using a linear classifier $h_{\boldsymbol{w}}(\mathbf{z})$. The following optimization problem is solved to maximize the likelihood of the data conditioned on the latent vector with the class-preserving regularization:

$$\max_{\theta,\varphi,\omega} \quad \mathbb{E}_{\mathbf{z}'} \log \boldsymbol{p}_\theta \left(\mathbf{x} \,\middle|\, \mathbf{z}' \right) \qquad (2)$$

$$\text{s.t. } Supp\big(\boldsymbol{q}_\varphi(\mathbf{z} \mid \mathbf{x} \in R_i)\big) \cap Supp\big(\boldsymbol{q}_\varphi(\mathbf{z} \mid \mathbf{x} \in R_j)\big) = \phi$$
$$\forall\, i \neq j \text{ where, } R_i = \{\mathbf{x} \mid h_{\boldsymbol{w}}(\mathbf{z}) = i\} \text{ and } h_{\boldsymbol{w}}(\mathbf{z}) = \boldsymbol{w}^T \mathbf{z}$$

The maximization of the conditional log-likelihood term in the above optimization problem ensures that the learned latent vectors ($\mathbf{z}$) are not de-generate. All the three networks, namely the encoder, decoder, and the classifier, are trained jointly with implicit minority oversampling by taking convex combinations of latent vectors corresponding to the same class during training. Upon convergence, the latent space (and hence the classifier) will be robust to class imbalance since they have been trained along with implicit class-preserving minority oversampling; this has been demonstrated empirically in the experiment section (c.f. Section 5). In the subsequent sections, we present implementation details and analyze our method with theoretical performance guarantees.

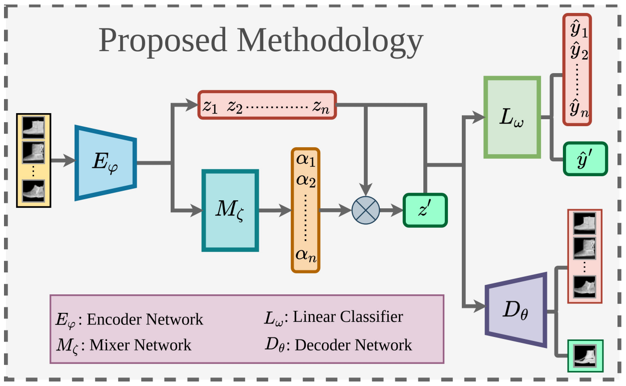


Figure 1: Architecture of the proposed methodology. It is a regularized autoencoder, where the latent space is regularized using a linear classifier to facilitate distance metric free class preserving oversampling of the minority classes. The decoder network maximizes the conditional data likelihood to avoid degeneracy in the latent space.

3.3 Implementation Details

Our model consists of an encoder network (E_φ) for $\boldsymbol{q}_\varphi(\mathbf{z} \mid \mathbf{x})$ and a decoder (D_θ) network for $\boldsymbol{p}_\theta(\mathbf{x} \mid \mathbf{z}')$; the latent space is regularized using a linear classifier (L_ω), which takes t-number of $\mathbf{z}_i$'s (from the same class) and the corresponding $\mathbf{z}'$ as inputs and predicts the corresponding label. The true label of $\mathbf{z}'$ is set to be the same as that of $\mathbf{z}_i$'s for class preservation as discussed earlier.

The mixing coefficients α used for oversampling (Eq. 1) can be chosen arbitrarily from any distribution, albeit, in Section 4, we argue that learning them leads to enhanced performance. To this end, we employ a Mixer network (M_ζ) which takes the latent vectors $\mathbf{z}_i$'s as input and gives α as output from a softmax layer. The mixer network is trained to produce the α which would make it hard for the linear classifier (L_ω) to correctly classify the sampled latent vector. To achieve this, the mixer net is made to minimize a cross-entropy loss which is calculated by keeping the label of $\mathbf{z}'$ different from that of $\mathbf{z}_i$'s. Intuitively, this would force the mixer net to generate samples (through α) that are 'difficult' for the linear classifier to classify. Formally, the loss for the mixer network $\mathcal{L}_{mixer}$ is:

$$\mathcal{L}_{mixer} = -\sum_{k=1}^{C} \mathbf{y}_j^{(k)} \log \hat{\mathbf{y}}'^{(k)} \qquad j \neq i \tag{3}$$

where, $\hat{\mathbf{y}}'^{(k)} = L_\omega(\mathbf{z}')^{(k)}$ denotes the k^{th} element of output of the linear classifier and $\mathbf{y}_j^{(k)}$ is the indicator of the true class. Note that, from Equation 1, $\hat{\mathbf{y}}'^{(k)}$ is a function of α and so is $\mathcal{L}_{mixer}$.

To train the decoder network, that seeks to maximize $\boldsymbol{p}_\theta(\mathbf{x} \mid \mathbf{z}')$, we use standard adversarial loss (Gulrajani *et al.*, 2017) between the output of the decoder ($\tilde{\mathbf{x}}$) and true data samples ($\mathbf{x}$). We seek a distributional match between the input data and decoder output but not a sample-by-sample match since data points corresponding to oversampled latent codes do not exist in the given dataset. This demands the use of a critic network denoted by C_ψ. We use the loss outlined in WGAN-GP (Gulrajani *et al.*, 2017) for its superior stability and convergence to train the the critic network. Accordingly, the loss functions for the decoder and critic networks are given by:

$$\mathcal{L}_{decoder} = -\mathop{\mathbb{E}}_{\tilde{\mathbf{x}} \sim P_G} [C_\psi(\tilde{\mathbf{x}})] \tag{4}$$

$$\mathcal{L}_{critic} = \mathop{\mathbb{E}}_{\tilde{\mathbf{x}} \sim P_G} [C_\psi(\tilde{\mathbf{x}})] - \mathop{\mathbb{E}}_{\mathbf{x} \sim P_{data}} [C_\psi(\mathbf{x})] + \lambda \mathop{\mathbb{E}}_{\hat{\mathbf{x}} \sim P_{\hat{\mathbf{x}}}} \left[(\|\nabla_{\hat{\mathbf{x}}} C_\psi(\hat{\mathbf{x}})\|_2 - 1)^2 \right] \tag{5}$$

where, P_G denotes the distribution of decoder's output and P_{data} denotes the marginal of true data distribution. $P_{\hat{\mathbf{x}}}$ is the distribution of points sampled uniformly along straight line between points sampled from the data distribution P_{data} and the decoder distribution P_G.

Next, the linear classifier (used for latent regularization and final classification) is trained via $(t+1)$ categorical crossentropy loss terms (one for $\mathbf{z}'$ and t for each of corresponding $\mathbf{z}_i$'s):

$$\mathcal{L}_{clf} = -\sum_{i=1}^{t} \sum_{j=1}^{C} \mathbf{y}_i^{(j)} \log \hat{\mathbf{y}}_i^{(j)} - \sum_{j=1}^{C} \mathbf{y}_i^{(j)} \log \hat{\mathbf{y}}'^{(j)} \tag{6}$$

where $\hat{\mathbf{y}}_i^{(j)} = L_\omega(\mathbf{z}_i)^{(j)}$, $\hat{\mathbf{y}}'^{(j)} = L_\omega(\mathbf{z}')^{(j)}$ are the class-wise outputs of the linear classifier for $\mathbf{z}_i$ and $\mathbf{z}'$, respectively and $\mathbf{y}_i^{(j)}$ is the indicator for the j^{th} true class for the $\mathbf{z}_i$. Lastly, the encoder network is using the above defined $\mathcal{L}_{clf}$ and an additional mean absolute error loss between the generated image from the decoder and the original image. Hence the encoder loss is given by:

$$\mathcal{L}_{encoder} = \lambda_{clf} \cdot \mathcal{L}_{clf} + \lambda_{mae} \cdot \mathcal{L}_{mae} \tag{7}$$

$$\text{where,} \quad \mathcal{L}_{mae} = \sum_i \|\mathbf{x}_i - D_\theta \odot E_\varphi(\mathbf{x}_i)\|_1$$

The overall block diagram of the proposed method can be found in Figure 1. For all our experiments, we have set $\lambda_{clf} = 5$ and $\lambda_{mae} = 0.01$. A detailed description of the training procedure is described in Algorithm 1 of the supplementary.

4 Theoretical Analysis

In this section, we analyze and derive theoretical guarantees on different parts of the proposed design. The proofs for all the propositions can be found in the Supplementary material.

The first proposition is to show that the constraint enforced in Eq. 2, preserves class labels. Without loss of generality, we show it for a binary classification problem.

Proposition 1 (Label Preservation)**.** *In a binary classification problem, let* $\mathcal{S}_0 = \{\mathbf{z}_i \mid y_i = C_0\}$ *and* $\mathcal{S}_1 = \{\mathbf{z}_i \mid y_i = C_1\}$. *Also suppose* $\mathcal{S}_0$ *and* $\mathcal{S}_1$ *are linearly separable. Then,* $\sum_{i=1}^{N_0} \alpha_i \mathbf{z}_{i \in \mathcal{S}_0} \in C_0$ *and* $\sum_{j=1}^{N_1} \alpha_j \mathbf{z}_{i \in \mathcal{S}_1} \in C_1$, *where* $\alpha_i, \alpha_j \in \mathbb{R}^+$ *and* $\sum_{i=1}^{N_0} \alpha_i = \sum_{j=1}^{N_1} \alpha_j = 1$.

From the above proposition, it is clear that if the data points from different classes are linearly separable (in some space $\mathcal{Z}$), then oversampling via convex combinations according to Eq. 1 preserves the class, in that particular space. Accordingly, in our method, we focus on learning a latent space that by construction induces linearly separable class-conditional distributions. Note that prior works (Zhang *et al.*, 2017a; Verma *et al.*, 2019; Berthelot *et al.*, 2018) take inter-class convex combination as opposed to intra-class convex combination, which does not ensure class preservation in the proposed setting.

Our next claim is that oversampling via convex combinations can result in a reduced variance in the empirical risk, which is desired for better generalization. Let us define, $\hat{R}(\ell; D)$ and $\hat{R}_\alpha(\ell; D)$ to be the empirical risk of a classifier without and with convex oversampling on a dataset $\mathcal{D} \sim \mathbb{P}$ of size N, respectively.

$$\hat{R}(\ell; D) = \frac{1}{N} \sum_i \ell(\mathbf{z}_i) \tag{8}$$

$$\hat{R}_\alpha(\ell; D) = \frac{1}{N} \sum_i \ell\left(\alpha^T \mathbf{z}^i\right) \tag{9}$$

where, $\alpha = [\alpha_1 \dots \alpha_n]^T$ such that $\sum_i \alpha_i = 1$ and $\mathbf{z}^i = [\mathbf{z}_1 \dots \mathbf{z}_n]^T$ where $\mathbf{z}_i$'s are sampled i.i.d from D and belong to same class. Then we have the following proposition:

Proposition 2. *For a convex function $\ell(\cdot) : \mathbb{R}^d \to \mathbb{R}^+$ whose variance is upper bounded by a constant B, there exists some α such that:*

$$\mathbb{V}_\mathbb{P}\left[\hat{R}_\alpha(\ell; D)\right] \leq \mathbb{V}_\mathbb{P}\left[\hat{R}(\ell; D)\right] \tag{10}$$

where $\mathbb{V}_\mathbb{P}[\cdot]$ is the variance operator.

Proposition 2 asserts that if the mixing coefficients are chosen appropriately, the variance of estimate of empirical risk for the oversampling case will be less than that of without oversampling. This can be used directly to give the following proposition.

Proposition 3. *Let $\ell(\cdot)$ denote a bounded loss function. Fix a hypothesis class $\mathcal{F}$ of predictors $f : \mathcal{Z} \to \mathbb{R}^C$, with induced class $\mathcal{H}^* \subset [0,1]^\mathcal{Z}$ of functions $h(\mathbf{z}) = \ell(\alpha^T \mathbf{z})$. Suppose $\mathcal{H}^*$ has uniform covering number $\mathcal{N}_\infty$. Then, for α chosen appropriately and any $\delta \in (0,1)$, with probability atleast $1 - \delta$ over $D \sim \mathbb{P}$,*

$$R(f) \leq \hat{R}_\alpha(\ell; D) + \mathcal{O}\left(\sqrt{\mathbb{V}_N^\alpha(\ell) \cdot \frac{\log \frac{\mathcal{M}_N^*}{\delta}}{N}} + \frac{\log \frac{\mathcal{M}_N^*}{\delta}}{N}\right) \tag{11}$$

where $\mathcal{M}_N^* = \mathcal{N}_\infty\left(\frac{1}{N}, \mathcal{H}^*, 2N\right)$ and $\mathbb{V}_N^\alpha$ is the empirical variance of the loss values $\{\ell(\alpha^T \mathbf{z}^i)\}_{i=1}^N$

Note that the given PAC bound is tight for the oversampling case as compared to the without oversampling case because of Proposition 2, making the risk estimate closer to the true risk. Additionally, both Prop. 2 and 3 are valid for a range of α (as seen in the proof of Prop. 2) which motivates the need for the mixer network to learn α in our method.

Next, we analyze the robustness of the proposed method in terms of the Lipschitz constant, specifically, a classifier with unbounded lipschitz constant is prone to overfitting and outliers, leading to high generalization errors. However, if the lipschitz constant of a classifier is bounded then the interpolations will be smooth and hence the the generalization error will be low as shown in the following proposition.

Proposition 4. *Consider a dataset $(\mathbf{x}_i, y_i)_{i=1}^N$ where $\mathbf{x}_i$ is i.i.d uniform on the sphere $\mathbb{S}^{d-1} = \left\{\mathbf{x} \in \mathbb{R}^d \mid \|\mathbf{x}\| = 1\right\}$, y_i is uniform on $\{-1, +1\}$ and $N \leq c \cdot d$ for some $c \in \mathbb{R}$. Let $h_\alpha^*(\mathbf{x}) = w_\alpha^T(\alpha \mathbf{x}_1 + (1-\alpha)\mathbf{x}_2)$ be the optimal linear classifier, where, $w_\alpha \in \mathbb{R}^d$ and $\mathbf{x}_1, \mathbf{x}_2$ have the same label. Then $h_\alpha^*(\mathbf{x})$ has an upper bound on the lipschitz constant.*

The assumptions required for this proposition to hold are same as those mentioned in Theorem 5.58 of Vershynin (2010). As note by them, the assumptions can be relaxed and extended to datasets where x_i are i.i.d from a centered Gaussian with covariance $\frac{1}{d} I_d$ and where y_i are i.i.d with random signs. This is similar to the definition of 'generic' datasets defined by Bubeck *et al.* (2021). We further note that the bounds obtained in this proposition hold for any dataset that satisfies the conditions of a generic dataset. Therefore, if one chooses to sample α from a fixed (non-learnable) distribution, whose support lies outside the given range, then the given bounds need not hold true.

Proposition 4 can be used to derive conditions under which the bound on lipschitz constant for the oversampling case will be tighter than that of without oversampling case. This idea is presented in the following Corollary.

Corollary 1. *Let N be the size of original dataset and M be the size of dataset after oversampling. Then for $N \leq M \leq (1+r)N$, there exists some α for which the bound on $Lip(h_\alpha^*(\mathbf{x}))$ is tighter than $Lip(h^*(\mathbf{x}))$ (lipschitz constant for without oversampling case) for some $r \in \mathbb{R}^+$.*

The above corollary tells us that if oversampled in right amount, the trained linear classifier will be more robust than the one trained without oversampling. The upper bound on number of oversampled points (M) can be interpreted as follows: naively sampling beyond a certain threshold leads to decrease in diversity of sampled points which will lead the classifier to overfit.

Thus in summary, we showed that our method which oversamples in a class-preserving manner does not only results in lower empirical risk and tighter PAC bounds, but also is robust to overfitting and outliers. We confirm our hypothesis by extensive experiments in the next section.

5 Experimental Results

5.1 Dataset Description and Baselines

To validate the efficacy of the proposed method, we consider the following five real-world datasets — MNIST(Lecun, 2010), Fashion MNIST (Xiao *et al.*, 2017), SVHN (Netzer *et al.*, 2011), CIFAR-10 (Krizhevsky, 2009), and CELEB-A (Liu *et al.*, 2015) as in the recent baselines (Mullick *et al.*, 2019; Wang *et al.*, 2020; Dablain *et al.*, 2022). Since these datasets are not significantly imbalanced, following the prior works (Mullick *et al.*, 2019; Wang *et al.*, 2020; Dablain *et al.*,

Table 1: Performance (Mean ± Std. Dev.) of our method on Balanced test dataset

Dataset	Metrices	Ours	DeepSMOTE (Dablain *et al.*, 2022)	GAMO (Mullick *et al.*, 2019)	BAGAN (Mariani *et al.*, 2018)	DGC (Wang *et al.*, 2020)
MNIST	ACSA	**96.79 ± 0.09**	92.60 ± 0.50	94.97 ± 0.32	92.76 ± 0.45	94.60 ± 0.31
	F1	**96.78 ± 0.09**	92.44 ± 0.54	94.90 ± 0.33	92.62 ± 0.49	94.53 ± 0.33
	GM	**98.20 ± 0.05**	95.83 ± 0.29	97.19 ± 0.18	95.92 ± 0.26	96.90 ± 0.21
FashionMNIST	ACSA	**84.79 ± 0.15**	82.78 ± 0.06	83.29 ± 0.40	79.93 ± 1.42	82.34 ± 0.38
	F1	**84.30 ± 0.18**	82.11 ± 0.11	82.63 ± 0.44	79.23 ± 1.74	81.48 ± 0.35
	GM	**91.30 ± 0.09**	90.11 ± 0.04	90.41 ± 0.24	88.40 ± 0.86	89.02 ± 0.21
SVHN	ACSA	**78.66 ± 0.47**	70.67 ± 0.31	74.60 ± 0.45	70.19 ± 1.13	68.91 ± 1.89
	F1	**78.42 ± 0.48**	69.64 ± 0.26	74.05 ± 0.49	70.85 ± 2.81	68.16 ± 1.95
	GM	**87.62 ± 0.29**	82.68 ± 0.20	85.14 ± 0.28	81.86 ± 0.17	80.77 ± 1.35
CIFAR10	ACSA	**57.94 ± 0.15**	41.51 ± 0.58	45.65 ± 0.73	43.35 ± 1.73	41.39 ± 1.12
	F1	**57.13 ± 0.15**	38.51 ± 0.52	43.05 ± 0.97	40.40 ± 1.47	39.75 ± 1.24
	GM	**74.31 ± 0.10**	62.30 ± 0.46	65.49 ± 0.55	63.73 ± 1.07	60.49 ± 1.01
CELEBA	ACSA	**74.65 ± 0.26**	62.70 ± 0.84	66.80 ± 0.29	66.22 ± 0.84	67.55 ± 0.56
	F1	**74.02 ± 0.32**	60.05 ± 1.03	65.09 ± 0.27	63.11 ± 0.23	65.62 ± 0.47
	GM	**83.67 ± 0.15**	75.39 ± 0.59	78.27 ± 0.20	77.57 ± 0.13	77.54 ± 0.33
CIFAR100	ACSA	**26.41 ± 0.15**	23.33 ± 0.08	12.68 ± 0.39	22.99 ± 0.70	24.64 ± 0.11
	F1	**23.99 ± 0.19**	22.19 ± 0.11	09.67 ± 0.30	22.74 ± 0.66	22.29 ± 0.10
	GM	**51.20 ± 0.11**	48.11 ± 0.08	35.45 ± 0.55	48.42 ± 0.90	45.76 ± 0.05
ImageNet100	ACSA	**25.15 ± 0.13**	12.66 ± 0.17	07.90 ± 0.28	17.16 ± 0.69	08.82 ± 0.12
	F1	**22.31 ± 0.11**	11.55 ± 0.14	06.44 ± 0.25	15.72 ± 0.32	07.64 ± 0.07
	GM	**44.81 ± 0.32**	35.42 ± 0.22	27.97 ± 0.48	41.25 ± 0.56	23.90 ± 0.01

Table 2: Statistical significance: p-values obtained from Shaffer post-hoc tests and Bayesian Wilcoxon signed-rank tests for pairwise comparison of the proposed method with the baseline oversampling-based approaches for the ACSA metric. We have combined the results from imbalanced and long-tailed recognition test cases.

Our vs.	Shaffer post-hoc	Bayesian Wilcoxon signed-rank
DeepSMOTE	5.15×10^{-14}	2.40×10^{-3}
GAMO	1.27×10^{-6}	1.15×10^{-2}
DGC	7.01×10^{-12}	2.96×10^{-2}
BAGAN	2.91×10^{-14}	1.80×10^{-4}

2022), we introduce imbalance by randomly sub-sampling disproportionate number of samples from different classes. As can be seen from Table 7 in the supplementary material, the imbalance ratio of the majority class and the smallest minority class is 100:1 for MNIST and Fashion MNIST. For SVHN, CIFAR-10, and CelebA, the imbalance ratio is approximately 56:1. For CelebA, following the prior works (Wang *et al.*, 2020; Dablain *et al.*, 2022), five non-overlapping classes (black hair, brown hair, blond, gray, and bald) were selected. Further to evaluate the proposed method's performance over a large number of classes we consider CIFAR-100 (Krizhevsky, 2009) (cf. Table 8 in supp.) and ImageNet-100 (previously used in (Kalantidis *et al.*, 2020; Van Gansbeke *et al.*, 2020; Ravula *et al.*, 2021)) (cf. Table 9 in supp.).

As in prior work (Wang *et al.*, 2020), two kinds of test datasets were considered. In the first setting, the ratio of test examples across different classes follow a similar ratio as present in the training split (Imbalanced Test data). In the second setting, we keep an equal amount of data across all the classes in the test split (Balanced Test data). We compare the performance of our method with several state-of-the-art deep resampling methods: BAGAN (Mariani *et al.*, 2018), GAMO (Mullick *et al.*, 2019), DGC (Wang *et al.*, 2020), and DeepSMOTE (Dablain *et al.*, 2022).
Additionally, we test our approach on tabular datasets (Alcalá-Fdez *et al.*, 2011) to see how it performs in a variety of settings. The results on tabular datasets are presented in the Table 13 of the supplementary material.

5.2 Performance Metrics

For quantitative evaluation of classification performance, it is imperative to select unbiased performance metrics for the majority or minority classes. Like previous works (Mullick *et al.*, 2019; Wang *et al.*, 2020; Dablain *et al.*, 2022), we consider the following three metrics: Average Class Specific Accuracy (ACSA), macro-averaged Geometric Mean (GM), and macro-averaged F1 score (F1) as these are not biased towards any specific class (Sokolova & Lapalme, 2009; Mullick *et al.*, 2020).
Further, to quantify the goodness of the learned latent space for tasks beyond minority classification, we study the quality of the images generated by the decoder from the learned

Table 3: Density/Coverage performance comparison of the baselines with our method

	MNIST		FashionMNIST		SVHN		CIFAR10		CelebA	
	Density	Coverage	Density	Coverage	Density	Coverage	Density	Coverage	Density	Coverage
BAGAN (Mariani *et al.*, 2018)	0.257	0.338	0.563	0.132	0.010	0.005	0.393	0.054	0.143	0.006
GAMO2PIX (Mullick *et al.*, 2019)	0.521	0.346	0.769	0.403	0.106	0.080	2.240	0.438	0.743	0.229
DeepSMOTE (Dablain *et al.*, 2022)	0.345	0.305	0.396	0.287	0.076	0.126	1.138	0.287	0.316	0.141
Ours (Proposed Method)	0.729	0.511	1.177	0.379	0.104	0.117	2.042	0.362	0.919	0.341

Table 4: Ablation Studies: Contribution of each component in classification performance.

Linear Classifier	Oversampling	Decoder	Mixer Network	CIFAR10			CELEBA		
				ACSA	GM	F1	ACSA	GM	F1
✗	✓	✓	✗	20.58	31.21	13.51	29.28	42.34	26.39
✓	✗	✓	✗	35.51	47.41	29.20	63.94	67.83	57.04
✓	✓	✗	✓	52.16	69.04	50.93	73.20	82.33	72.86
✓	✓	✓	✗	54.51	71.94	53.07	74.38	83.45	73.98
✓	✓	✓	✓	57.94	74.31	57.13	74.65	83.67	74.02

Table 5: ACSA of the proposed method at various values of bottleneck layer dimensionality, m.

Dataset	m	ACSA		
		$m/4$	$m/2$	$2*m$
MNIST	64	97.23	97.44	97.53
FMNIST	64	85.70	85.82	86.04
SVHN	128	80.11	80.75	80.26
CIFAR10	256	57.64	57.20	56.44
CelebA	256	75.98	75.50	74.32

latent space. Note that this is not the primary objective of our method since the goal is not to learn a generative model. Nevertheless, we report density and coverage metrics (Naeem *et al.*, 2020) between the output of the decoder and the true data, to quantify quality and diversity, respectively. Density is unbounded. A higher density score indicates better quality. Coverage is bounded by 1, and a higher coverage score indicates better diversity. Additionally, we report the Fréchet Inception Distance (FID) (Heusel *et al.*, 2017) between the true data samples and outputs of our decoder in Table 11 of the supplementary material.

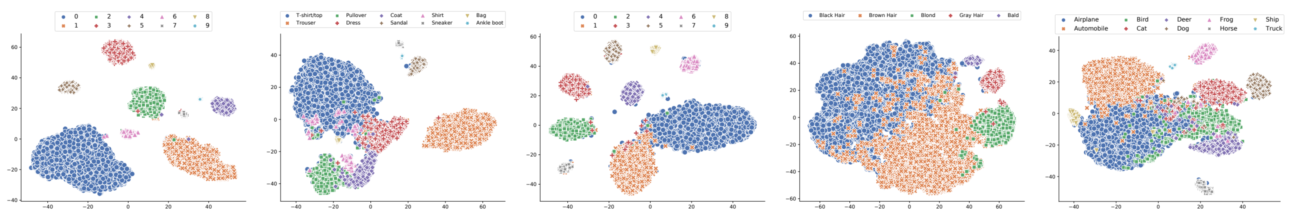
5.3 Model Architecture and Compute Resource

The proposed method uses a small Convolution-based architecture for Encoder, Discriminator, and Transpose Convolution-based architecture for the Decoder network. These architectures are derived from prior work (Dai & Wipf, 2019; Mondal *et al.*, 2021). The output layer of the linear classifier contains neurons equal to the number of classes in the dataset. The mixer network is a simple multi-layer perceptron (MLP). Refer to the supplementary material (Table 16, Table 17) for details of the architecture. For baseline methods, either exact or equal capacity architectures were used. The proposed method, GAMO (Mullick *et al.*, 2019), and DGC (Wang *et al.*, 2020) are one-stage frameworks with a built-in classifier. BAGAN (Mariani *et al.*, 2018) and DeepSMOTE (Dablain *et al.*, 2022) first augment the minority classes and then train a standalone classifier on the balanced data. Following (Dablain *et al.*, 2022), for these two methods we use a Resnet-18 architecture (He *et al.*, 2016) for the classifier. For all of our experiments, we have chosen $t = 2$, as the performance as measured by ACSA is similar for $t = 2, 3$ and 4 (see Figure 3 in the supplementary).

All the experiments in this paper, including the hyper-parameter search and reproducing the baseline methods, were conducted on a machine with Intel® Xeon® Silver 4216 CPU (@ 2.10 GHz), 128GB RAM and 48GB NVIDIA RTX A6000 GPU.

5.4 Results

In Table 1 we compare our method with other oversampling methods (mentioned in Section 5.1) on Balanced test dataset. Refer to the supplementary material Table 10 for performance on imbalanced test set. For each method, we compute and report the mean and standard deviation (error) over three runs. As can be seen, majority of the time, single-stage methods (GAMO and DGC) exhibit superior performance over two-stage methods (BAGAN and DeepSMOTE). This might be ascribed to the fact that the samples generated using two-stage methods are not guaranteed to be useful for learning the decision boundary among the classes. Finally, the proposed method outperforms all of the baselines by a significant margin on all the considered metrics. Similar comparisons and observations are made with the classical oversampling methods as seen in



(a) MNIST (b) FashionMNIST (c) SVHN (d) CelebA (e) CIFAR10

Figure 2: t-SNE visualization of the latent space.

Tables 14 and 15 of the supplementary. Further, following Dablain *et al.* (2022), we conduct the Shaffer post-hoc test (Stapor *et al.*, 2021) and the Bayesian Wilcoxon signed-rank test (Benavoli *et al.*, 2017) for statistical comparison over several datasets to determine if the proposed method outperforms the baseline resampling techniques. Both the tests utilise a statistical significance level of 0.05. Table 2 presents the statistical test results for the ACSA metric. It is seen that the performance boost achieved by the proposed method is statistically significant.
The density and coverage metrics for the output of the decoder are mentioned in Table 3. Refer to the supplementary material Table 11 for FID comparison and output samples. It is noteworthy that, our method achieves best sample quality in all but 1-2 cases. This suggests that the proposed method not only aids classification but also produces good quality samples under the presence of class-imbalance.

In Figure 2, we visualise the linear separability of the latent space learnt by our method through t-SNE plot. It is observed that the performance of the proposed method corroborates the linear separability assumption. For example, for simple datasets such as MNIST the classes are well separated and the proposed method achieves almost perfect classification score. On the other hand, for complex datasets such as CIFAR-10 the classes are overlapping in the latent space and therefore we observe a dip in the classification accuracy. However, our performance is still significantly better than the SOTA.

5.5 Ablation Studies

To examine the impact of each component in the proposed method, we conduct several ablation studies. As can be seen from Table 4, when the linear classifier and mixer network are not present, the performance is the worst. This might be ascribed to the fact that the oversampling is not guaranteed to preserve the class label anymore. In the absence of decoder (class preserving regularization is imposed on the output of the encoder without the decoder) or the mixer network (uniform sampling for α), the performance degrades albeit not as severely without oversampling. This suggests that the most important component is the class-preserving convex oversampling. However, learning α and/or avoiding latent degeneration provides considerable boost especially in a 'difficult' dataset such as CIFAR-10, CIFAR-100 and ImageNet-100. In Table 5, we study the impact of the bottleneck layer's dimensionality (m) on classification performance. It is seen that the performance as measured using ACSA remains almost constant over a wide range of m. Next, we validate the effectiveness of learning the mixing coefficients (α) as opposed to sampling them according to some fixed distribution. For example, when $\alpha \sim Beta(2, 2)$, ACSA achieved on CIFAR10 is 52.35, which is better than the peroformance obtained using no class-preserving interpolation but still worse than the case of learnable α.

6 Conclusion, Limitations, Risks, and Broader Impact

Real-world data frequently exhibit skewed distributions with a long tail rather than the ideal uniform distributions across each class. There are numerous ways to deal with this problem. We seek to improve upon existing oversampling approaches by developing a label-preserving oversampling strategy. We have also included theoretical guarantees for the suggested method. We anticipate that practitioners will be able to work with significantly skewed data in real-world circumstances by using this strategy. However, data in many real-world applications like autonomous driving, medical diagnostics, and healthcare may impose additional limitations on the learning process and final models, such as being fair or private, in addition to being innately unbalanced. Our method, albeit theoretical sound and empirically observed to perform, warrants rigorous validations in the presence of the aforementioned additional constraints in critical, high-stakes applications before deployment.

Acknowledgements

We thank IIT Delhi HPC facility[1] for computational resources. Prathosh AP would like to acknowledge grant received from Tvarit GmbH, Ludwigstraße 31, 60327 Frankfurt am Main, Germany. Parag Singla is supported by IBM AI Horizon Networks (AIHN) grant and IBM SUR awards.

[1]`http://supercomputing.iitd.ac.in`

Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of the funding agencies.

References

Abdi, Lida, & Hashemi, Sattar. 2016. To Combat Multi-Class Imbalanced Problems by Means of Over-Sampling Techniques. *IEEE Transactions on Knowledge and Data Engineering*, **28**(1), 238–251.

Alcalá-Fdez, Jesús, Fernández, Alberto, Luengo, Julián, Derrac, Joaquín, García, Salvador, Sánchez, Luciano, & Herrera, Francisco. 2011. Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, **17**.

Ando, Shin, & Huang, Chun Yuan. 2017. Deep over-sampling framework for classifying imbalanced data. *Pages 770–785 of: Joint European Conference on Machine Learning and Knowledge Discovery in Databases*.

Arjovsky, Martin, Chintala, Soumith, & Bottou, Léon. 2017. Wasserstein Generative Adversarial Networks. *Pages 214–223 of: Proc. of ICML*, vol. 70.

Barandela, Ricardo, Valdovinos, Rosa Maria, & Sánchez, José Salvador. 2003. New applications of ensembles of classifiers. *Pattern Analysis & Applications*, **6**(3), 245–256.

Batista, Gustavo EAPA, Prati, Ronaldo C, & Monard, Maria Carolina. 2004. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, **6**(1), 20–29.

Benavoli, Alessio, Corani, Giorgio, Demšar, Janez, & Zaffalon, Marco. 2017. Time for a Change: a Tutorial for Comparing Multiple Classifiers Through Bayesian Analysis. *Journal of Machine Learning Research*, **18**(77), 1–36.

Berthelot, David, Raffel, Colin, Roy, Aurko, & Goodfellow, Ian. 2018. Understanding and improving interpolation in autoencoders via an adversarial regularizer. *arXiv preprint arXiv:1807.07543*.

Błaszczyński, Jerzy, Deckert, Magdalena, Stefanowski, Jerzy, & Wilk, Szymon. 2010. Integrating selective pre-processing of imbalanced data with ivotes ensemble. *Pages 148–157 of: International conference on rough sets and current trends in computing*.

Bubeck, Sébastien, Li, Yuanzhi, & Nagaraj, Dheeraj M. 2021. A law of robustness for two-layers neural networks. *Pages 804–820 of: Conference on Learning Theory*.

Cao, Kaidi, Wei, Colin, Gaidon, Adrien, Arechiga, Nikos, & Ma, Tengyu. 2019. Learning imbalanced datasets with label-distribution-aware margin loss. *In: Proc. of NeurIPS*, vol. 32.

Chawla, Nitesh V, Bowyer, Kevin W, Hall, Lawrence O, & Kegelmeyer, W Philip. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, **16**, 321–357.

Chawla, Nitesh V, Lazarevic, Aleksandar, Hall, Lawrence O, & Bowyer, Kevin W. 2003. SMOTEBoost: Improving prediction of the minority class in boosting. *Pages 107–119 of: European conference on principles of data mining and knowledge discovery*.

Chen, Junya, Xiu, Zidi, Goldstein, Benjamin, Henao, Ricardo, Carin, Lawrence, & Tao, Chenyang. 2021. Supercharging Imbalanced Data Learning With Energy-based Contrastive Representation Transfer. *Proc. of NeurIPS*, **34**.

Chen, Xi, Duan, Yan, Houthooft, Rein, Schulman, John, Sutskever, Ilya, & Abbeel, Pieter. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *In: Proc. of NeurIPS*.

Chu, Peng, Bian, Xiao, Liu, Shaopeng, & Ling, Haibin. 2020. Feature space augmentation for long-tailed data. *Pages 694–710 of: European Conference on Computer Vision*.

Cui, Yin, Jia, Menglin, Lin, Tsung-Yi, Song, Yang, & Belongie, Serge. 2019. Class-balanced loss based on effective number of samples. *Pages 9268–9277 of: Proc. of CVPR*.

Dablain, Damien, Krawczyk, Bartosz, & Chawla, Nitesh V. 2022. DeepSMOTE: Fusing deep learning and SMOTE for imbalanced data. *IEEE Transactions on Neural Networks and Learning Systems*.

Dai, Bin, & Wipf, David. 2019. Diagnosing and enhancing vae models. *In: Proc. of ICLR*.

Das, Swagatam, Mullick, Sankha Subhra, & Zelinka, Ivan. 2022. On Supervised Class-Imbalanced Learning: An Updated Perspective and Some Key Challenges. *IEEE Transactions on Artificial Intelligence*, **3**(6), 973–993.

Deng, Jia, Dong, Wei, Socher, Richard, Li, Li-Jia, Li, Kai, & Fei-Fei, Li. 2009. ImageNet: A large-scale hierarchical image database. *Pages 248–255 of: 2009 IEEE Conference on Computer Vision and Pattern Recognition*.

Domingos, Pedro. 1999. MetaCost: A General Method for Making Classifiers Cost-Sensitive. *Page 155–164 of: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Dong, Qi, Gong, Shaogang, & Zhu, Xiatian. 2018. Imbalanced deep learning by minority class incremental rectification. *IEEE transactions on pattern analysis and machine intelligence*, **41**(6), 1367–1381.

Douzas, Georgios, & Bacao, Fernando. 2018. Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert Systems with applications*, **91**, 464–471.

Fan, Wei, Stolfo, Salvatore J., Zhang, Junxin, & Chan, Philip K. 1999. AdaCost: Misclassification Cost-Sensitive Boosting. *Page 97–105 of: Proc. of ICML.*

Freund, Yoav, & Schapire, Robert E. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, **55**(1), 119–139.

Galar, Mikel, Fernández, Alberto, Barrenechea, Edurne, & Herrera, Francisco. 2013. EUSBoost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling. *Pattern recognition*, **46**(12), 3460–3471.

García, Salvador, & Herrera, Francisco. 2009. Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evolutionary computation*, **17**(3), 275–306.

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, & Bengio, Yoshua. 2014. Generative Adversarial Nets. *In: Proc. of NeuRIPS.*

Gulrajani, Ishaan, Ahmed, Faruk, Arjovsky, Martin, Dumoulin, Vincent, & Courville, Aaron C. 2017. Improved training of wasserstein gans. *Advances in neural information processing systems*, **30**.

Guo, Hongyu, & Viktor, Herna L. 2004. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *ACM Sigkdd Explorations Newsletter*, **6**(1), 30–39.

Han, Hui, Wang, Wen-Yuan, & Mao, Bing-Huan. 2005. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. *Pages 878–887 of: International conference on intelligent computing*. Springer.

Hart, P. 1968. The condensed nearest neighbor rule (Corresp.). *IEEE Transactions on Information Theory*, **14**(3), 515–516.

He, Haibo, Bai, Yang, Garcia, Edwardo A, & Li, Shutao. 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. *Pages 1322–1328 of: 2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence).*

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, & Sun, Jian. 2016. Deep residual learning for image recognition. *Pages 770–778 of: Proceedings of the IEEE conference on computer vision and pattern recognition.*

Heusel, Martin, Ramsauer, Hubert, Unterthiner, Thomas, Nessler, Bernhard, & Hochreiter, Sepp. 2017. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. *Pages 6626–6637 of:* Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., & Garnett, R. (eds), *Advances in Neural Information Processing Systems 30.*

Hu, Zhiting, Tan, Bowen, Salakhutdinov, Russ R, Mitchell, Tom M, & Xing, Eric P. 2019. Learning data manipulation for augmentation and weighting. *Proc. of NeurIPS*, **32**.

Huang, Chen, Li, Yining, Loy, Chen Change, & Tang, Xiaoou. 2016. Learning deep representation for imbalanced classification. *Pages 5375–5384 of: Proc. of CVPR.*

Huang, Gao, Liu, Zhuang, Van Der Maaten, Laurens, & Weinberger, Kilian Q. 2017. Densely connected convolutional networks. *Pages 4700–4708 of: Proceedings of the IEEE conference on computer vision and pattern recognition.*

Kalantidis, Yannis, Sariyildiz, Mert Bulent, Pion, Noe, Weinzaepfel, Philippe, & Larlus, Diane. 2020. Hard negative mixing for contrastive learning. *Proc. of NeurIPS*, **33**, 21798–21809.

Kang, Bingyi, Xie, Saining, Rohrbach, Marcus, Yan, Zhicheng, Gordo, Albert, Feng, Jiashi, & Kalantidis, Yannis. 2020. Decoupling representation and classifier for long-tailed recognition. *In: Proc. of ICLR.*

Karakoulas, Grigoris, & Shawe-Taylor, John. 1998. Optimizing Classifiers for Imbalanced Training Sets. *Page 253–259 of: Proc. of NeurIPS.*

Kim, Jaehyung, Hur, Youngbum, Park, Sejun, Yang, Eunho, Hwang, Sung Ju, & Shin, Jinwoo. 2020. Distribution aligning refinery of pseudo-label for imbalanced semi-supervised learning. *Proc. of NeurIPS*, **33**, 14567–14579.

Kingma, Diederik P, & Welling, Max. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114.*

Kini, Ganesh Ramachandra, Paraskevas, Orestis, Oymak, Samet, & Thrampoulidis, Christos. 2021. Label-imbalanced and group-sensitive classification under overparameterization. *Advances in Neural Information Processing Systems*, **34**.

Koziarski, Michał. 2020. Radial-based undersampling for imbalanced data classification. *Pattern Recognition*, **102**, 107262.

Koziarski, Michał, Krawczyk, Bartosz, & Woźniak, Michał. 2019. Radial-based oversampling for noisy imbalanced data classification. *Neurocomputing*, **343**, 19–33.

Koziarski, Michał, Woźniak, Michał, & Krawczyk, Bartosz. 2020. Combined cleaning and resampling algorithm for multi-class imbalanced data with label noise. *Knowledge-Based Systems*, **204**, 106223.

Krizhevsky, Alex. 2009. *Learning multiple layers of features from tiny images*. Tech. rept.

Kubat, Miroslav, & Matwin, Stan. 1997. Addressing the Curse of Imbalanced Training Sets: One-Sided Selection. *Pages 179–186 of: ICML.*

Kubat, Miroslav, Holte, Robert C, & Matwin, Stan. 1998. Machine learning for the detection of oil spills in satellite radar images. *Machine learning*, **30**(2), 195–215.

Laurikkala, Jorma. 2001. Improving identification of difficult small classes by balancing class distribution. *Pages 63–66 of: Conference on artificial intelligence in medicine in Europe*. Springer.

Le, Lei, Patterson, Andrew, & White, Martha. 2018. Supervised autoencoders: Improving generalization performance with unsupervised regularizers. *Advances in neural information processing systems*, **31**.

Lecun, Y. 2010. *The mnist database of handwritten digits*. `http://yann.lecun.com/exdb/mnist/`.

Lee, Hae Beom, Lee, Hayeon, Na, Donghyun, Kim, Saehoon, Park, Minseop, Yang, Eunho, & Hwang, Sung Ju. 2020. Learning to Balance: Bayesian Meta-Learning for Imbalanced and Out-of-distribution Tasks. *In: Proc. of ICLR.*

Lee, Hyuck, Shin, Seungjae, & Kim, Heeyoung. 2021. ABC: Auxiliary Balanced Classifier for Class-imbalanced Semi-supervised Learning. *In:* Beygelzimer, A., Dauphin, Y., Liang, P., & Vaughan, J. Wortman (eds), *Proc. of NeurIPS.*

Lemaître, Guillaume, Nogueira, Fernando, & Aridas, Christos K. 2017. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *The Journal of Machine Learning Research*, **18**(1), 559–563.

Li, Buyu, Liu, Yu, & Wang, Xiaogang. 2019. Gradient harmonized single-stage detector. *Pages 8577–8584 of: Proc. of AAAI*, vol. 33.

Li, Mingchen, Zhang, Xuechen, Thrampoulidis, Christos, Chen, Jiasi, & Oymak, Samet. 2021. AutoBalance: Optimized Loss Functions for Imbalanced Data. *Advances in Neural Information Processing Systems*, **34**.

Li, Zeju, Kamnitsas, Konstantinos, & Glocker, Ben. 2020. Analyzing overfitting under class imbalance in neural networks for image segmentation. *IEEE transactions on medical imaging*, **40**(3), 1065–1077.

Lin, Tsung-Yi, Goyal, Priya, Girshick, Ross, He, Kaiming, & Dollár, Piotr. 2017a. Focal loss for dense object detection. *Pages 2980–2988 of: Proc. of ICCV.*

Lin, Wei-Chao, Tsai, Chih-Fong, Hu, Ya-Han, & Jhang, Jing-Shang. 2017b. Clustering-based undersampling in class-imbalanced data. *Information Sciences*, **409**, 17–26.

Liu, Xu-Ying, Wu, Jianxin, & Zhou, Zhi-Hua. 2008. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, **39**(2), 539–550.

Liu, Zhining, Wei, Pengfei, Jiang, Jing, Cao, Wei, Bian, Jiang, & Chang, Yi. 2020a. MESA: boost ensemble imbalanced learning with meta-sampler. *Advances in Neural Information Processing Systems*, **33**, 14463–14474.

Liu, Zhining, Cao, Wei, Gao, Zhifeng, Bian, Jiang, Chen, Hechang, Chang, Yi, & Liu, Tie-Yan. 2020b. Self-paced ensemble for highly imbalanced massive data classification. *Pages 841–852 of: 2020 IEEE 36th international conference on data engineering (ICDE)*. IEEE.

Liu, Ziwei, Luo, Ping, Wang, Xiaogang, & Tang, Xiaoou. 2015. Deep Learning Face Attributes in the Wild. *In: Proc. of ICCV.*

Lucic, Mario, Kurach, Karol, Michalski, Marcin, Bousquet, Olivier, & Gelly, Sylvain. 2018. Are GANs Created Equal? A Large-scale Study. *In: Proc. of NeuRIPS.*

Makhzani, Alireza, Shlens, Jonathon, Jaitly, Navdeep, & Goodfellow, Ian. 2016. Adversarial Autoencoders. *In: Proc. of ICLR.*

Mani, Inderjeet, & Zhang, I. 2003. kNN approach to unbalanced data distributions: a case study involving information extraction. *Pages 1–7 of: Proceedings of workshop on learning from imbalanced datasets*, vol. 126. ICML.

Mariani, Giovanni, Scheidegger, Florian, Istrate, Roxana, Bekas, Costas, & Malossi, Cristiano. 2018. Bagan: Data augmentation with balancing gan. *arXiv preprint arXiv:1803.09655.*

Maurer, Andreas, & Pontil, Massimiliano. 2009. Empirical bernstein bounds and sample variance penalization. *arXiv preprint arXiv:0907.3740.*

Mirza, Mehdi, & Osindero, Simon. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784.*

Mondal, Arnab Kumar, Chowdhury, Sankalan Pal, Jayendran, Aravind, Singla, Parag, Asnani, Himanshu, & Prathosh, AP. 2020. MaskAAE: Latent space optimization for Adversarial Auto-Encoders. *In: Proc. of UAI.*

Mondal, Arnab Kumar, Asnani, Himanshu, Singla, Parag, & Prathosh, AP. 2021. FlexAE: Flexibly Learning Latent Priors for Wasserstein Auto-Encoders. *In: Proc. of UAI.*

Mullick, Sankha Subhra, Datta, Shounak, & Das, Swagatam. 2019. Generative adversarial minority oversampling. *Pages 1695–1704 of: Proc. of ICCV.*

Mullick, Sankha Subhra, Datta, Shounak, Dhekane, Sourish Gunesh, & Das, Swagatam. 2020. Appropriateness of performance indices for imbalanced data classification: An analysis. *Pattern Recognition*, **102**, 107197.

Naeem, Muhammad Ferjad, Oh, Seong Joon, Uh, Youngjung, Choi, Yunjey, & Yoo, Jaejun. 2020. Reliable Fidelity and Diversity Metrics for Generative Models. *In: Proc. of ICML.*

Netzer, Yuval, Wang, Tao, Coates, Adam, Bissacco, Alessandro, Wu, Bo, & Ng, Andrew Y. 2011. Reading digits in natural images with unsupervised feature learning. *In: NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Nguyen, Hien M, Cooper, Eric W, & Kamei, Katsuari. 2011. Borderline over-sampling for imbalanced data classification. *International Journal of Knowledge Engineering and Soft Data Paradigms*, **3**(1), 4–21.

Park, Seulki, Lim, Jongin, Jeon, Younghan, & Choi, Jin Young. 2021. Influence-balanced loss for imbalanced visual classification. *Pages 735–744 of: Proc. of ICCV*.

Ramentol, Enislay, Caballero, Yailé, Bello, Rafael, & Herrera, Francisco. 2012. SMOTE-RSB*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using SMOTE and rough sets theory. *Knowledge and information systems*, **33**(2), 245–265.

Rao, R Bharat, Krishnan, Sriram, & Niculescu, Radu Stefan. 2006. Data mining for improved cardiac care. *Acm Sigkdd Explorations Newsletter*, **8**(1), 3–10.

Ravula, Sriram, Smyrnis, Georgios, Jordan, Matt, & Dimakis, Alexandros G. 2021. Inverse Problems Leveraging Pre-trained Contrastive Representations. *Proc. of NeurIPS*, **34**, 8753–8765.

Sáez, José A, Luengo, Julián, Stefanowski, Jerzy, & Herrera, Francisco. 2015. SMOTE–IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. *Information Sciences*, **291**, 184–203.

Santurkar, Shibani, Schmidt, Ludwig, & Madry, Aleksander. 2018. A Classification-Based Study of Covariate Shift in GAN Distributions. *Pages 4480–4489 of: Proceedings of the 35th International Conference on Machine Learning*, vol. 80.

Sarafianos, Nikolaos, Xu, Xiang, & Kakadiaris, Ioannis A. 2018. Deep imbalanced attribute classification using visual attention aggregation. *Pages 680–697 of: Proceedings of the European Conference on Computer Vision (ECCV)*.

Seiffert, Chris, Khoshgoftaar, Taghi M, Van Hulse, Jason, & Napolitano, Amri. 2009. RUSBoost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, **40**(1), 185–197.

Shahbazi, Mohamad, Danelljan, Martin, Paudel, Danda Pani, & Gool, Luc Van. 2022. Collapse by Conditioning: Training Class-conditional GANs with Limited Data. *In: International Conference on Learning Representations*.

Shalev-Shwartz, Shai, & Ben-David, Shai. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Shu, Jun, Xie, Qi, Yi, Lixuan, Zhao, Qian, Zhou, Sanping, Xu, Zongben, & Meng, Deyu. 2019. Meta-Weight-Net: Learning an explicit mapping for sample weighting. *Proc. of NeurIPS*, **32**.

Sokolova, Marina, & Lapalme, Guy. 2009. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, **45**(4), 427–437.

Stapor, Katarzyna, Ksieniewicz, Paweł, García, Salvador, & Woźniak, Michał. 2021. How to design the fair experimental classifier evaluation. *Applied Soft Computing*, **104**, 107219.

Stefanowski, Jerzy, & Wilk, Szymon. 2008. Selective preprocessing of imbalanced data for improving classification performance. *Pages 283–292 of: International Conference on Data Warehousing and Knowledge Discovery*.

Ting, Kai Ming. 2002. An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering*, **14**(3).

Tolstikhin, Ilya, Bousquet, Olivier, Gelly, Sylvain, & Scholkopf, Bernhard. 2018. Wasserstein Auto-Encoders. *In: Proc. of ICLR*.

Tomek, Ivan. 1976. Two modifications of CNN. *IEEE Trans. Systems, Man and Cybernetics*, **6**, 769–772.

Van Gansbeke, Wouter, Vandenhende, Simon, Georgoulis, Stamatios, Proesmans, Marc, & Van Gool, Luc. 2020. Scan: Learning to classify images without labels. *Pages 268–285 of: Proc. of ECCV*.

Verma, Vikas, Lamb, Alex, Beckham, Christopher, Najafi, Amir, Mitliagkas, Ioannis, Lopez-Paz, David, & Bengio, Yoshua. 2019. Manifold mixup: Better representations by interpolating hidden states. *Pages 6438–6447 of: International Conference on Machine Learning*. PMLR.

Vershynin, Roman. 2010. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*.

Viola, Paul, & Jones, Michael. 2002. Fast and Robust Classification using Asymmetric AdaBoost and a Detector Cascade. *Proc. of NeurIPS*, **14**(04).

Vuttipittayamongkol, Pattaramon, & Elyan, Eyad. 2020. Neighbourhood-based undersampling approach for handling imbalanced and overlapped data. *Information Sciences*, **509**, 47–70.

Wang, Shoujin, Liu, Wei, Wu, Jia, Cao, Longbing, Meng, Qinxue, & Kennedy, Paul J. 2016. Training deep neural networks on imbalanced data sets. *Pages 4368–4374 of: 2016 international joint conference on neural networks (IJCNN)*.

Wang, Shuo, & Yao, Xin. 2009. Diversity analysis on imbalanced data sets by using ensemble models. *Pages*

324–331 of: IEEE symposium on computational intelligence and data mining.

Wang, Shuo, & Yao, Xin. 2012. Multiclass imbalance problems: Analysis and potential solutions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, **42**(4), 1119–1130.

Wang, Xinyue, Lyu, Yilin, & Jing, Liping. 2020 (June). Deep Generative Model for Robust Imbalance Classification. *In: Proc. of CVPR*.

Wang, Yiru, Gan, Weihao, Yang, Jie, Wu, Wei, & Yan, Junjie. 2019. Dynamic curriculum learning for imbalanced data classification. *Pages 5017–5026 of: Proc. of ICCV*.

Wang, Yu-Xiong, Ramanan, Deva, & Hebert, Martial. 2017. Learning to model the tail. *Proc. of NeurIPS*, **30**.

Wei, Wei, Li, Jinjiu, Cao, Longbing, Ou, Yuming, & Chen, Jiahang. 2013. Effective detection of sophisticated online banking fraud on extremely imbalanced data. *World Wide Web*, **16**(4), 449–475.

Wilson, Dennis L. 1972. Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE Transactions on Systems, Man, and Cybernetics*, **SMC-2**(3), 408–421.

Wu, Lijun, Tian, Fei, Xia, Yingce, Fan, Yang, Qin, Tao, Jian-Huang, Lai, & Liu, Tie-Yan. 2018. Learning to teach with dynamic loss functions. *Proc. of NeurIPS*, **31**.

Xiao, Han, Rasul, Kashif, & Vollgraf, Roland. 2017. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*.

Yang, Xuebing, Kuang, Qiuming, Zhang, Wensheng, & Zhang, Guoping. 2017. AMDO: An over-sampling technique for multi-class imbalanced problems. *IEEE Transactions on Knowledge and Data Engineering*, **30**(9), 1672–1685.

Yang, Yuzhe, & Xu, Zhi. 2020. Rethinking the value of labels for improving class-imbalanced learning. *Proc. of NeurIps*, **33**, 19290–19301.

Yi, Xin, Walia, Ekta, & Babyn, Paul. 2019. Generative adversarial network in medical imaging: A review. *Medical Image Analysis*, **58**, 101552.

Yuan, Xiaohui, Xie, Lijun, & Abouelenien, Mohamed. 2018. A regularized ensemble framework of deep learning for cancer detection from multi-class, imbalanced training data. *Pattern Recognition*, **77**, 160–172.

Zhang, Hongyi, Cisse, Moustapha, Dauphin, Yann N, & Lopez-Paz, David. 2017a. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Zhang, Xiao, Fang, Zhiyuan, Wen, Yandong, Li, Zhifeng, & Qiao, Yu. 2017b. Range loss for deep face recognition with long-tailed training data. *Pages 5409–5418 of: Proc. of ICCV*.

Zhao, Caidan, & Lei, Yang. 2021. Intra-class cutmix for unbalanced data augmentation. *Pages 246–251 of: 2021 13th International Conference on Machine Learning and Computing*.

Zheng, Zeyu, Oh, Junhyuk, & Singh, Satinder. 2018. On learning intrinsic rewards for policy gradient methods. *Advances in Neural Information Processing Systems*, **31**.

Zhou, Zhi-Hua, & Liu, Xu-Ying. 2006. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, **18**(1), 63–77.

A Theoretical Results

In this section, we analyze our method and give some guarantees on different estimates. For notations and definitions, please refer to Section 4 of the main text.
The first proposition is the key statement which we use to enforce the constraint in Eq. 2 of the main paper.

Proposition 5 (Label Preservation)**.** *In a binary classification problem, let $\mathcal{S}_0 = \{\mathbf{z}_i \mid y_i = C_0\}$ and $\mathcal{S}_1 = \{\mathbf{z}_i \mid y_i = C_1\}$. Also let $\mathcal{S}_0$ and $\mathcal{S}_1$ are linearly separable. Then, $\sum_{i=1}^{N_0} \alpha_i \mathbf{z}_{i\in\mathcal{S}_0} \in C_0$ and $\sum_{j=1}^{N_1} \alpha_j \mathbf{x}_{i\in\mathcal{S}_1} \in C_1$, where $\alpha_i, \alpha_j \in \mathbb{R}^+$ and $\sum_{i=1}^{N_0} \alpha_i = \sum_{j=1}^{N_1} \alpha_j = 1$.*

Proof. Since the datapoints are linearly separable $\exists\, W$ and $\boldsymbol{b}$ such that

$$W^T\mathbf{z} + \boldsymbol{b} < 0,\ \forall \mathbf{z} \in \mathcal{S}_0 \tag{12}$$

$$W^T\mathbf{z} + \boldsymbol{b} > 0,\ \forall \mathbf{z} \in \mathcal{S}_1 \tag{13}$$

Now,

$$W^T \sum_{i=1}^{N_0} \alpha_i \mathbf{z}_{i\in\mathcal{S}_0} + \boldsymbol{b} = \sum_{i=1}^{N_0} \alpha_i W^T \mathbf{z}_{i\in\mathcal{S}_0} + \sum_{i=1}^{N_0} \alpha_i \boldsymbol{b} = \sum_{i=1}^{N_0} \alpha_i (W^T \mathbf{z}_{i\in\mathcal{S}_0} + \boldsymbol{b}) < 0 \tag{14}$$

Similarly, it can be shown,

$$W^T \sum_{j=1}^{n_1} \alpha_j \mathbf{z}_{i\in\mathcal{S}_1} + \boldsymbol{b} > 0 \tag{15}$$

Therefore, novel datapoint resulting out of the convex combination of datapoints of the same class preserves the class label. □

Proposition 6. *For a convex function $\ell(\cdot) : \mathbb{R}^d \to \mathbb{R}^+$ whose variance is upper bounded by a constant B, there exists some α such that:*

$$\mathbb{V}_{\mathbb{P}}\left[\hat{R}_\alpha(\ell; D)\right] \le \mathbb{V}_{\mathbb{P}}\left[\hat{R}(\ell; D)\right] \tag{16}$$

where $\mathbb{V}_{\mathbb{P}}[\cdot]$ is the variance operator.

Proof. By definition, we have:

$$\begin{aligned}
\mathbb{V}_{\mathbb{P}}\left[\hat{R}(\ell; D)\right] &= \mathbb{V}_{\mathbb{P}}\left[\frac{1}{N}\sum_i \ell(\mathbf{z}_i)\right] \\
&= \frac{1}{N^2}\sum_i \mathbb{V}_{\mathbb{P}}\left[\ell(\mathbf{z}_i)\right] \\
&= \frac{1}{N}\mathbb{V}_{\mathbb{P}}\left[\ell(\mathbf{z})\right]
\end{aligned}$$

where, we have used the fact that $\mathbf{z}_i$'s are iid and hence, $\mathbb{V}_{\mathbb{P}}\left[\ell(\mathbf{z}_i)\right] = \mathbb{V}_{\mathbb{P}}\left[\ell(\mathbf{z})\right]\ \forall i$. Now, similarly:

$$\begin{aligned}
\mathbb{V}_{\mathbb{P}}\left[\hat{R}_\alpha(\ell; D)\right] &= \mathbb{V}_{\mathbb{P}}\left[\frac{1}{N}\sum_i \ell\left(\alpha^T\mathbf{z}^i\right)\right] \\
&= \frac{1}{N^2}\sum_i \mathbb{V}_{\mathbb{P}}\left[\ell\left(\alpha^T\mathbf{z}^i\right)\right] \\
&= \frac{1}{N}\mathbb{V}_{\mathbb{P}}\left[\ell\left(\alpha^T\mathbf{z}^i\right)\right]
\end{aligned}$$

Using $\mathbb{V}[\cdot]$ and $\mathbb{E}[\cdot]$ for brevity, consider the following:

$$\begin{aligned}
\mathbb{V}_{\mathbb{P}}\left[\ell\left(\alpha^T\mathbf{z}^i\right)\right] &= \mathbb{E}\left[\ell(\alpha^T\mathbf{z})^2\right] - \left(\mathbb{E}\left[\ell(\alpha^T\mathbf{z})\right]\right)^2 \\
&\leq \mathbb{E}\left[\left(\sum_i \alpha_i \ell(\mathbf{z}_i)\right)^2\right] - \left(\mathbb{E}\left[\ell(\alpha^T\mathbf{z})\right]\right)^2 \qquad (\because \ell(\cdot) \text{ is convex}) \\
&= \mathbb{E}\left[\sum_i \alpha_i^2 \ell(\mathbf{z}_i)^2 + 2\sum_{i,j}\alpha_i\alpha_j\ell(\mathbf{z}_i)\ell(\mathbf{z}_j)\right] - \left(\mathbb{E}\left[\ell(\alpha^T\mathbf{z})\right]\right)^2 - \sum_i \alpha_i^2\left(\mathbb{E}[\ell(\mathbf{z}_i)]\right)^2 + \sum_i \alpha_i^2\left(\mathbb{E}[\ell(\mathbf{z}_i)]\right)^2 \\
&= \|\alpha\|_2^2\mathbb{V}[\ell(\mathbf{z})] + \left(\mathbb{E}[\ell(\mathbf{z})]\right)^2 - \left(\mathbb{E}\left[\ell\left(\sum_i \alpha_i\mathbf{z}_i\right)\right]\right)^2 \\
&\leq \|\alpha\|_2^2\mathbb{V}[\ell(\mathbf{z})] + \left(\mathbb{E}[\ell(\mathbf{z})]\right)^2 - \left(\ell\left(\sum_i \alpha_i\mathbb{E}[\mathbf{z}]\right)\right)^2 \qquad (\because \text{Jensen's Inequality}) \\
&= \|\alpha\|_2^2\mathbb{V}[\ell(\mathbf{z})] + \left(\mathbb{E}[\ell(\mathbf{z})] - \ell(\mathbb{E}[\mathbf{z}])\right)\left(\mathbb{E}[\ell(\mathbf{z})] + \ell(\mathbb{E}[\mathbf{z}])\right)
\end{aligned}$$

Now, the second term in the above expression is non-negative by Jensen' Inequality and definition of $\ell(\cdot)$. Let, $\left(\mathbb{E}[\ell(\mathbf{z})] - \ell(\mathbb{E}[\mathbf{z}])\right)\left(\mathbb{E}[\ell(\mathbf{z})] + \ell(\mathbb{E}[\mathbf{z}])\right) = K \geq 0$. Therefore, we have:

$$\mathbb{V}\left[\ell\left(\alpha^T\mathbf{z}^i\right)\right] \leq \|\alpha\|_2^2\mathbb{V}[\ell(\mathbf{z})] + K$$

Since, $\mathbb{V}[\ell(\mathbf{z})] \leq B$, we have for $\{\alpha \mid \|\alpha\|_2^2 \leq 1 - \frac{K}{B}\}$:

$$\begin{aligned}
\mathbb{V}\left[\ell\left(\alpha^T\mathbf{z}\right)\right] &\leq \|\alpha\|_2^2\mathbb{V}[\ell(\mathbf{z})] + K \leq \mathbb{V}\left[\ell(\mathbf{z})\right] \\
\implies \mathbb{V}_{\mathbb{P}}\left[\hat{R}_\alpha(\ell; D)\right] &\leq \frac{1}{N}\mathbb{V}_{\mathbb{P}}\left[\ell\left(\alpha^T\mathbf{z}^i\right)\right] \\
&\leq \frac{1}{N}\mathbb{V}_{\mathbb{P}}\left[\ell(\mathbf{z})\right] = \mathbb{V}_{\mathbb{P}}\left[\hat{R}(\ell; D)\right]
\end{aligned}$$

□

Proposition 7. *For a bounded loss function $\ell(\cdot)$. Fix a hypothesis class $\mathcal{F}$ of predictors $f : \mathcal{X} \to \mathbb{R}^C$, with induced class $\mathcal{H}^* \subset [0,1]^{\mathcal{X}}$ of functions $h(\mathbf{x}) = \ell(\alpha^T\mathbf{x})$. Suppose $\mathcal{H}^*$ has uniform covering number $\mathcal{N}_\infty$. Then, for α chosen appropriately and any $\delta \in (0,1)$, with probability atleast $1-\delta$ over $D \sim \mathbb{P}$,*

$$R(f) \leq \hat{R}_\alpha(\ell; D) + \mathcal{O}\left(\sqrt{\mathbb{V}_N^\alpha(\ell)\cdot\frac{\log\frac{\mathcal{M}_N^*}{\delta}}{N}} + \frac{\log\frac{\mathcal{M}_N^*}{\delta}}{N}\right) \tag{17}$$

where $\mathcal{M}_N^* = \mathcal{N}_\infty\left(\frac{1}{N}, \mathcal{H}^*, 2N\right)$ and $\mathbb{V}_N^\alpha$ is the empirical variance of the loss values $\{\ell(\alpha^T\mathbf{x}^i)\}_{i=1}^N$

Proof. The proof follows directly from Proposition 6 and a version of Bennet's equality from (Maurer & Pontil, 2009) [Theorem 6]. □

Proposition 8. *Consider a dataset $(\mathbf{x}_i, y_i)_{i=1}^N$ where $\mathbf{x}_i$ is i.i.d uniform on the sphere $\mathbb{S}^{d-1} = \left\{\mathbf{x} \in \mathbb{R}^d \mid \|\mathbf{x}\| = 1\right\}$, y_i is uniform on $\{-1, +1\}$ and $N \leq c \cdot d$ for some $c \in \mathbb{R}$. Let $h_\alpha^*(\mathbf{x}) = w_\alpha^T(\alpha\mathbf{x}_1 + (1-\alpha)\mathbf{x}_2)$ where, $w_\alpha \in \mathbb{R}^d$ and $\mathbf{x}_1, \mathbf{x}_2$ have the same label be the optimal linear classifier. Then $h_\alpha^*(\mathbf{x})$ has an upper bound on the lipschitz constant.*

Proof. The general solution to the system $Xw = Y$ where X is $N \times d$ matrix with i^{th} row equal to $\mathbf{x}_i$ and $Y = (y_1, y_2, \ldots, y_N)$ is:

$$w = X^T(XX^T)^{-1}Y$$

However, for the oversampling case, the matrix X will become $\bar{X} = \alpha X + (1-\alpha)PX$ where P is any permutation matrix such that $PY = Y$. Hence, the solution to above optimization problem is:

$$w_\alpha = \bar{X}^T(\bar{X}\bar{X}^T)^{-1}Y$$

Now, using (Vershynin, 2010) [Theorem 5.58] and $N \leq c \cdot d$, we have with probability atleast $1 - \exp{(C - cd)}$:

$$XX^T \succeq \frac{1}{2} I_N$$

Consider the following:

$$\begin{aligned}\bar{X}\bar{X}^T &= \alpha^2 XX^T + (1-\alpha)^2 PXX^TP^T + \\ &\quad \alpha(1-\alpha)\left[XX^TP^T + PXX^T\right] \\ &\succeq \frac{\alpha^2}{2} I_N + \frac{(1-\alpha)^2}{2}(PP^T) + \frac{\alpha(1-\alpha)}{2}(P^T + P)\end{aligned}$$

Now, since P is a permutation matrix, we have $PP^T = I_N$ and $(P^T + P)$ is symmetric $\implies (P^T + P) \succeq kI_N$ where $k = \lambda_{min}(P^T + P)$ is the minimum eigen value of $P^T + P$. Hence, we have the following:

$$\begin{aligned}\bar{X}\bar{X}^T &\succeq \frac{\alpha^2 + (1-\alpha)^2 + k\alpha(1-\alpha)}{2} I_N \\ &= \frac{1}{2} d(\alpha) I_N \\ &\quad \text{where, } d(\alpha) = \alpha^2 + (1-\alpha)^2 + k\alpha(1-\alpha) \\ \implies \left(\bar{X}\bar{X}^T\right)^{-1} &\preceq \frac{2}{d(\alpha)} I_N\end{aligned}$$

Let $Lip(h^*_\alpha(x))$ denote the lipschitz constant of $h^*_\alpha(x)$, then we have:

$$Lip(h^*_\alpha(x)) = \|w\| = \sqrt{Y^T\left(\bar{X}\bar{X}^T\right)^{-1}Y} \leq \sqrt{\frac{2N}{d(\alpha)}}$$

□

Corollary 2. *Let M be the size of original dataset and N be the size of dataset after oversampling. Then for $M \leq N \leq (1+r)M$, there exists some α for which the bound on $Lip(h^*_\alpha(\mathbf{x}))$ is tighter than $Lip(h^*(\mathbf{x}))$ (lipschitz constant for without oversampling case) for some $r \in \mathbb{R}^+$.*

Proof. For the above conditions, we have from previous Proposition:

$$Lip(h^*_\alpha(x)) \leq \sqrt{\frac{2N}{d(\alpha)}} \quad \text{and} \quad Lip(h^*(x)) \leq \sqrt{2M}$$

For a tight bound, we have:

$$\begin{aligned}&\sqrt{\frac{2N}{d(\alpha)}} \leq \sqrt{2M} \\ &\implies \frac{N}{\alpha^2 + (1-\alpha)^2 + k\alpha(1-\alpha)} \leq M \\ &\implies \alpha^2 - \alpha + \frac{N-M}{M(k-2)} \leq 0 \\ &\implies \alpha \in \left[\frac{1-\sqrt{1-\frac{4(N-M)}{M(k-2)}}}{2}, \frac{1-\sqrt{1+\frac{4(N-M)}{M(k-2)}}}{2}\right]\end{aligned}$$

For above range to be valid, we need:

$$1 \geq \frac{4(N-M)}{M(k-2)} \implies N \leq \left(1 + \frac{(k-2)}{4}\right) M$$

We know that $N \geq M$, thus we have:

$$M \leq N \leq (1+r)M \quad \text{where, } r = \frac{(k-2)}{4}$$

□

Algorithm 1: Proposed Method's Pseudo Code

```
Input: Imbalanced Dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$ with $C$ classes, Encoder Network $E_\varphi$, Decoder Network $D_\theta$, Linear
       Classifier $L_\omega$, Mixer Network $M_\zeta$ and a Critique Network $C_\psi$
Output: Trained Classifier $L_{\omega^*}$
Create $S_c = \{\mathbf{x}_i \mid y_i = c\}$
Create $S_{imb} = \bigcup_i S_{\{i:|S_i|<\max_j(|S_j|)}$
while not converged do
    for $(\mathbf{x}_i, y_i) \in \mathcal{D}$ do
        Sample $\{\mathbf{x}_{ij} \in S_{y_i}\}_{j=1}^{t-1}$ and let $\mathbf{x}_{it} = \mathbf{x}_i$
        Find latent vectors $\{\mathbf{z}_{ij} = E_{\varphi^*}(\mathbf{x}_{ij})\}_{j=1}^{t}$
        Find mixing coefficients $\{\alpha_{ij} = M_{\zeta^*}(\mathbf{z}_{ij})\}_{j=1}^{t}$
        Sample $\mathbf{z}' = \sum_{j=1}^{t} \alpha_{ij}\mathbf{z}_{ij}$
        Calculate the predicted labels $\hat{y}' = L_{\omega^*}(\mathbf{z}')$ and $\{\hat{y}_{ij} = L_{\omega^*}(\mathbf{z}_{ij})\}$
        Reconstruct the images $\{\hat{\mathbf{x}}_{ij} = D_{\theta^*}(\mathbf{z}_{ij})\}_{j=1}^{t}$ and $\hat{\mathbf{x}}' = D_{\theta^*}(\mathbf{z}')$
        Calculate $\mathcal{L}_{mixer}$, $\mathcal{L}_{decoder}$, $\mathcal{L}_{critic}$, $\mathcal{L}_{clf}$, $\mathcal{L}_{encoder}$ according to Eq. (3-7) of main paper.
        $\omega^* \leftarrow \omega^* - \gamma_1 \nabla_{\omega^*} \mathcal{L}_{clf}$
        $\zeta^* \leftarrow \zeta^* - \gamma_2 \nabla_{\zeta^*} \mathcal{L}_{mixer}$
        $\varphi^* \leftarrow \varphi^* - \gamma_3 \nabla_{\varphi^*} (\mathcal{L}_{encoder}$
        $\theta^* \leftarrow \theta^* - \gamma_3 \nabla_{\theta^*} \mathcal{L}_{decoder}$
        $\psi^* \leftarrow \psi^* - \gamma_4 \nabla_{\psi^*} \mathcal{L}_{critic}$
    Return the learned classifier $L_{\omega^*}$
```

Table 6: Summarized Description of Benchmark Datasets

	Dimension $(h \times w \times c)$	# of Classes	Train Split Size	Test Split Size
MNIST (Lecun, 2010)	$28 \times 28 \times 1$	10	$60,000$	$10,000$
Fashion-MNIST (Xiao *et al.*, 2017)	$28 \times 28 \times 1$	10	$60,000$	$10,000$
SVHN (Netzer *et al.*, 2011)	$32 \times 32 \times 3$	10	$73,257$	$26,032$
CIFAR-10 (Krizhevsky, 2009)	$32 \times 32 \times 3$	10	$50,000$	$10,000$
CELEBA (Liu *et al.*, 2015)	$32 \times 32 \times 3$	5	$15,160$	$5,000$
CIFAR-100 (Krizhevsky, 2009)	$32 \times 32 \times 3$	100	$50,000$	$10,000$
ImageNet-100 (Deng *et al.*, 2009)	$32 \times 32 \times 3$	100	$50,973$	$5,000$

Table 7: Class distributions of the datasets used in this work for benchmarking.

Class	MNIST/Fashion MNIST			SVHN/CIFAR-10			CelebA		
	Train	Balanced Test	Imbalanced Test	Train	Balanced Test	Imbalanced Test	Train	Balanced Test	Imbalanced Test
0	4000	800	1000	4500	800	1000	9000	1000	1000
1	2000	800	500	2000	800	500	4500	1000	500
2	1000	800	250	1000	800	250	1000	1000	111
3	750	800	187	800	800	187	500	1000	55
4	500	800	125	600	800	125	160	1000	17
5	350	800	87	500	800	87			
6	200	800	50	400	800	50			
7	100	800	25	250	800	25			
8	60	800	15	150	800	15			
9	40	800	10	80	800	10			

A.1 Dataset Description

We use six image datasets to benchmark the proposed method.

1. MNIST (Lecun, 2010): MNIST dataset contains 60,000 gray-scale training images and 10,000 gray-scale test examples of handwritten digits.

Table 8: Class distribution for CIFAR-100

Training Set	Balanced Test Set	Imbalanced Test Set
500, 488, 477, 466, 455, 445, 434, 424, 415, 405,	80, 80, 80, 80, 80, 80, 80, 80, 80, 80,	100, 97, 95, 93, 91, 89, 86, 84, 83, 81,
396, 387, 378, 369, 361, 352, 344, 336, 328, 321,	80, 80, 80, 80, 80, 80, 80, 80, 80, 80,	79, 77, 75, 73, 72, 70, 68, 67, 65, 64,
314, 306, 299, 292, 286, 279, 273, 266, 260, 254,	80, 80, 80, 80, 80, 80, 80, 80, 80, 80,	62, 61, 59, 58, 57, 55, 54, 53, 52, 50,
248, 243, 237, 232, 226, 221, 216, 211, 206, 201,	80, 80, 80, 80, 80, 80, 80, 80, 80, 80,	49, 48, 47, 46, 45, 44, 43, 42, 41, 40,
197, 192, 188, 183, 179, 175, 171, 167, 163, 159,	80, 80, 80, 80, 80, 80, 80, 80, 80, 80,	39, 38, 37, 36, 35, 35, 34, 33, 32, 31,
156, 152, 149, 145, 142, 139, 135, 132, 129, 126,	80, 80, 80, 80, 80, 80, 80, 80, 80, 80,	31, 30, 29, 29, 28, 27, 27, 26, 25, 25,
123, 121, 118, 115, 112, 110, 107, 105, 102, 100,	80, 80, 80, 80, 80, 80, 80, 80, 80, 80,	24, 24, 23, 23, 22, 22, 21, 21, 20, 20,
98, 95, 93, 91, 89, 87, 85, 83, 81, 79,	80, 80, 80, 80, 80, 80, 80, 80, 80, 80,	19, 19, 18, 18, 17, 17, 17, 16, 16, 15,
77, 75, 74, 72, 70, 69, 67, 66, 64, 63,	80, 80, 80, 80, 80, 80, 80, 80, 80, 80,	15, 15, 14, 14, 14, 13, 13, 13, 12, 12,
61, 60, 58, 57, 56, 54, 53, 52, 51, 50	80, 80, 80, 80, 80, 80, 80, 80, 80, 80	12, 12, 11, 11, 11, 10, 10, 10, 10, 10

Table 9: Class distribution for Imagenet-100

Training Set	Balanced Test Set	Imbalanced Test Set
1300, 1270, 1240, 1212, 1184, 1157, 1130, 1104, 1079, 1054,	50, 50, 50, 50, 50, 50, 50, 50, 50, 50,	50, 48, 47, 46, 45, 44, 43, 42, 41, 40,
1030, 1006, 983, 960, 938, 917, 896, 875, 855, 835,	50, 50, 50, 50, 50, 50, 50, 50, 50, 50,	39, 38, 37, 36, 36, 35, 34, 33, 32, 32,
816, 797, 779, 761, 743, 726, 710, 693, 677, 662,	50, 50, 50, 50, 50, 50, 50, 50, 50, 50,	31, 30, 29, 29, 28, 27, 27, 26, 26, 25,
647, 632, 617, 603, 589, 575, 562, 549, 537, 524,	50, 50, 50, 50, 50, 50, 50, 50, 50, 50,	24, 24, 23, 23, 22, 22, 21, 21, 20, 20,
512, 500, 489, 478, 467, 456, 445, 435, 425, 415,	50, 50, 50, 50, 50, 50, 50, 50, 50, 50,	19, 19, 18, 18, 17, 17, 17, 16, 16, 15,
406, 397, 387, 378, 370, 361, 353, 345, 337, 329,	50, 50, 50, 50, 50, 50, 50, 50, 50, 50,	15, 15, 14, 14, 14, 13, 13, 13, 12, 12,
322, 314, 307, 300, 293, 286, 280, 273, 267, 261,	50, 50, 50, 50, 50, 50, 50, 50, 50, 50,	12, 12, 11, 11, 11, 11, 10, 10, 10, 10,
255, 249, 243, 237, 232, 227, 221, 216, 211, 206,	50, 50, 50, 50, 50, 50, 50, 50, 50, 50,	9, 9, 9, 9, 8, 8, 8, 8, 8, 7,
202, 197, 193, 188, 184, 180, 175, 171, 167, 164,	50, 50, 50, 50, 50, 50, 50, 50, 50, 50,	7, 7, 7, 7, 7, 6, 6, 6, 6, 6,
160, 156, 152, 149, 146, 142, 139, 136, 133, 130	50, 50, 50, 50, 50, 50, 50, 50, 50, 50	6, 6, 5, 5, 5, 5, 5, 5, 5, 5

2. Fashion-MNIST (Xiao *et al.*, 2017): Fashion MNIST is a drop-in replacement of the MNIST (Lecun, 2010) dataset. It consists of Zalando's article images.

3. SVHN (Netzer *et al.*, 2011): Street View House Number dataset consists of small, cropped digits from house numbers in Google Street View images. It has 73,257 images for training and 26,032 images for testing.

4. CIFAR-10 (Krizhevsky, 2009): The CIFAR-10 dataset includes 60,000 small colour images categorised into ten classes, each with 6,000 images. This database has a standard split of 50,000 training images and 10,000 testing images.

5. CelebA (Liu *et al.*, 2015): CelebFaces Attributes Dataset (CelebA) is a large-scale face attributes dataset of $202,599$ celebrity photos, each with 40 attribute annotations. We resize the images to 32×32 following previous work in imbalance learning (Dablain *et al.*, 2022).

6. CIFAR-100 (Krizhevsky, 2009): This dataset is a sub-class of the CIFAR-10 dataset, and comprises of 100 classes, each with 600 images. Each class has 500 training images and 100 testing images.

7. ImageNet-100 (Deng *et al.*, 2009): Following previous literature (Kalantidis *et al.*, 2020; Van Gansbeke *et al.*, 2020; Ravula *et al.*, 2021)), we randomly subsample 100 classes from ImageNet dataset to validate the proposed methods effectiveness on large scale dataset.

The key information about the datasets used in this paper is summarised in Table 6. Since, the class distributions are not skewed in the original datasets, we artificailly create imbalance following prior works (Mullick *et al.*, 2019; Dablain *et al.*, 2022; Wang *et al.*, 2020). Table 7 and 8 presents the training and test distribution used for experimentation.

Further, we use four tabular datasets (Alcalá-Fdez *et al.*, 2011) to validate the performance of the proposed method. We use the standard train-test split as provided by the Imbalanced-learn package (Lemaître *et al.*, 2017).

1. OPTICAL DIGITS (Alcalá-Fdez *et al.*, 2011): Majority vs minority class ratio for this dataset is 9.1:1. Total number of samples is 5,620 and size of feature vector is 64.

2. ISOLET (Alcalá-Fdez *et al.*, 2011): Majority vs minority class ratio for this dataset is 12:1. Total number of samples is 7,797 and size of feature vector is 617.

3. WEBPAGE (Alcalá-Fdez *et al.*, 2011): Majority vs minority class ratio for this dataset is 33:1. Total number of samples is 34,780 and size of feature vector is 300.

4. PROTEIN HOMO (Alcalá-Fdez *et al.*, 2011): Majority vs minority class ratio for this dataset is 11:1. Total number of samples is 145,751 and size of feature vector is 74.

A.2 Experimental Results

A.2.1 Performance of Deep Learning Methods

In Table 1 of the main paper, we have compared the performance of the proposed method on balanced testset. Here we provide results on imbalanced testset in Table 10. Further Table 11 compares sample quality of each method as measured by FID (Heusel *et al.*, 2017). Moreover, in Table 12 we also compare our solution with baseline methods on AURPC-OVA metric. AURPC-OVA refers to the multi-class version of Area under the Precision Recall Curve (AURPC) following One vs All (OVA) startegy. It has been shown by Mullick *et al.* (2020) that AURPC-OVA metric satifies all the desirable properties to be a suitable evaluation metric for classifiers under multi-class imbalance setting.

Table 10: Performance (Mean ± Std. Dev.) of our method on Imbalanced test dataset.

Dataset	**Metrices**	**Our**	**DeepSMOTE** (Dablain *et al.*, 2022)	**GAMO** (Mullick *et al.*, 2019)	**BAGAN** (Mariani *et al.*, 2018)	**DGC** (Wang *et al.*, 2020)
MNIST	ACSA	**96.75 ± 0.86**	93.30 ± 1.20	95.38 ± 0.82	93.26 ± 0.06	94.89 ± 0.35
	F1	94.74 ± 0.98	94.71 ± 0.58	**96.10 ± 0.52**	94.59 ± 0.33	94.61 ± 0.71
	GM	**98.29 ± 0.45**	96.51 ± 0.61	97.60 ± 0.42	96.67 ± 0.36	97.26 ± 0.19
FashionMNIST	ACSA	**86.24 ± 1.24**	82.81 ± 1.40	81.85 ± 2.44	80.56 ± 0.80	81.78 ± 0.71
	F1	**83.38 ± 1.22**	82.12 ± 2.10	80.47 ± 2.30	81.79 ± 0.65	80.97 ± 0.80
	GM	**92.30 ± 0.68**	90.44 ± 0.78	89.93 ± 1.40	89.20 ± 0.44	89.43 ± 0.35
SVHN	ACSA	**79.49 ± 2.04**	71.34 ± 3.69	75.18 ± 1.67	71.19 ± 0.36	69.28 ± 0.90
	F1	66.83 ± 1.34	69.15 ± 3.33	**72.77 ± 0.39**	71.49 ± 2.63	63.64 ± 0.71
	GM	**88.35 ± 1.17**	83.69 ± 2.22	86.13 ± 0.96	83.44 ± 0.10	81.24 ± 1.02
CIFAR10	ACSA	**56.63 ± 2.95**	42.46 ± 2.25	45.99 ± 2.14	43.47 ± 2.72	39.99 ± 0.32
	F1	**45.06 ± 1.44**	41.16 ± 1.93	44.52 ± 2.30	42.19 ± 2.22	37.12 ± 0.67
	GM	**73.84 ± 1.98**	63.79 ± 1.72	66.58 ± 1.60	64.60 ± 2.07	59.67 ± 0.08
CELEBA	ACSA	**76.85 ± 0.59**	63.28 ± 2.18	66.80 ± 0.54	65.91 ± 0.74	65.03 ± 0.76
	F1	**68.64 ± 2.66**	65.46 ± 1.40	67.11 ± 0.53	68.56 ± 0.81	63.72 ± 0.52
	GM	**84.23 ± 2.04**	77.12 ± 1.47	79.63 ± 0.33	80.36 ± 0.89	77.19 ± 1.02
CIFAR100	ACSA	**25.86 ± 0.17**	23.05 ± 0.23	12.14 ± 0.43	23.09 ± 0.46	23.41 ± 0.16
	F1	20.96 ± 0.29	23.02 ± 0.26	11.68 ± 0.36	**23.54 ± 0.55**	21.56 ± 0.10
	GM	**50.66 ± 0.17**	47.83 ± 0.23	34.70 ± 0.62	48.96 ± 0.69	42.03 ± 0.08
ImageNet100	ACSA	**26.21 ± 0.12**	12.85 ± 0.39	08.56 ± 0.16	15.55 ± 0.27	09.55 ± 0.35
	F1	**24.43 ± 0.28**	12.53 ± 0.31	08.41 ± 0.12	14.73 ± 0.29	08.65 ± 0.41
	GM	**44.14 ± 0.08**	35.70 ± 0.19	29.12 ± 0.67	39.27 ± 0.47	22.81 ± 1.20

Table 11: FID of samples obtained from oversampling

Method	MNIST	Fashion	SVHN	CIFAR-10	CelebA
BAGAN	28.72	75.05	201.82	133.82	184.33
GAMO2PIX	19.32	36.84	104.73	101.52	73.08
DeepSMOTE	13.27	42.39	113.92	86.40	45.57
Proposed Method	13.24	33.95	100.88	89.53	60.27

Table 12: Comparison of mAURPC-OVA

	MNIST	Fashion	SVHN	CIFAR10	CelebA
Our	0.9971	0.9233	0.8929	0.6389	0.8452
DeepSMOTE	0.9862	0.8821	0.8087	0.4477	0.7042
GAMO	0.9802	0.8906	0.8319	0.5004	0.7454

Table 13: Performance of our method on Tabular datasets.

Dataset	Metrices	Our	DeepSMOTE (Dablain *et al.*, 2022)	GAMO (Mullick *et al.*, 2019)	BAGAN (Mariani *et al.*, 2018)	DGC (Wang *et al.*, 2020)
OPTICAL DIGITS	ACSA	98.01	95.61	98.19	93.29	96.65
	F1	97.94	94.18	98.18	94.36	95.38
	GM	98.01	95.61	98.19	93.29	96.63
ISOLET	ACSA	95.69	92.98	93.75	90.65	90.97
	F1	97.77	91.86	93.72	91.29	89.03
	GM	95.66	92.99	93.75	90.66	90.71
WEBPAGE	ACSA	94.83	83.66	89.79	62.36	88.61
	F1	96.89	86.18	89.68	68.02	90.72
	GM	94.82	83.67	89.79	62.36	87.91
PROTEIN HOMO	ACSA	94.70	82.23	91.89	81.08	86.09
	F1	97.51	88.48	91.87	88.07	91.39
	GM	94.68	82.23	91.89	81.08	84.96

A.2.2 Performance of Classical Approaches

In this section we present the classification performance of the following classical machine learning methods: SMOTE (Chawla *et al.*, 2002), AMDO (Yang *et al.*, 2017), MC-CCR (Koziarski *et al.*, 2020), and MC-RBO (Koziarski *et al.*, 2019). These results have been taken from DeepSMOTE (Dablain *et al.*, 2022) and presented here for easy reference.

Table 14: Performance of Classical Oversampling methods on Imbalanced Test Dataset

Datasets	Metrics	Our	SMOTE	AMDO	MC-CCR	MC-RBO
MNIST	ACSA	96.75	81.48	84.29	86.19	87.25
	F1	94.74	82.44	84.88	86.46	88.69
	GM	98.29	83.99	88.73	92.04	94.46
FashionMNIST	ACSA	86.24	67.94	74.90	78.58	80.06
	F1	83.38	67.12	75.39	79.03	80.14
	GM	92.30	74.84	80.89	86.17	88.02
SVHN	ACSA	79.49	70.18	71.94	72.01	74.20
	F1	66.83	71.80	73.06	80.94	74.91
	GM	88.35	76.33	78.52	74.26	82.97
CIFAR10	ACSA	56.63	28.02	31.19	32.83	33.01
	F1	45.06	29.58	32.44	33.91	35.83
	GM	73.84	50.08	53.99	56.68	59.15
CELEBA	ACSA	76.85	60.29	63.54	65.23	67.11
	F1	68.64	60.03	62.94	64.88	80.52
	GM	84.23	70.48	72.86	77.14	65.37

A.3 Ablation on t:

We have chosen number of mixing components, $t = 2, 3, 4$ and performed experiments to compute ACSA on the benchmark datasets using our method. It can be seen in Figure 3, the performance achieved is more or less similar. So for all of our experiments, we have chosen $t = 2$.

A.4 Qualitative Samples

In the main paper, the performance of the proposed method is evaluated mainly quantitatively, using standard metrics: ACSA, macro-averaged GM, macro-averaged F1 score, and density/coverage (Naeem *et al.*, 2020) score. Further, in Section A.2 of the supplementary material we have presented more quantitative metrics for imbalanced test set and compared FID. We have used 800 generated samples and 800 real test examples from each class for computation of FID and density/coverage score for all datasets. It has been observed that the proposed method not only outperform all other current state-of-the-art models in classification task as measured using those metrics (ACSA, GM, F1) but also achieve competitive scores for generative task (either outperforms or comparable). In this section, we present qualitative results (generated

Table 15: Performance of Classical Oversampling methods on Balanced Test Dataset

Datasets	Metrics	Our	SMOTE	AMDO	MC-CCR	MC-RBO
MNIST	ACSA	96.79	87.98	88.34	90.83	91.28
	F1	96.78	85.02	87.28	91.22	92.49
	GM	98.20	89.99	91.03	93.18	94.62
FashionMNIST	ACSA	84.79	70.58	72.98	75.78	76.91
	F1	84.30	68.06	71.53	74.39	75.92
	GM	91.30	76.39	79.36	81.04	82.14
SVHN	ACSA	78.66	68.19	71.59	74.29	75.38
	F1	78.42	64.28	68.47	72.49	73.52
	GM	87.62	74.48	79.13	81.62	81.98
CIFAR10	ACSA	57.94	27.93	31.85	33.48	39.17
	F1	57.13	25.10	30.04	32.88	40.37
	GM	74.31	42.81	48.19	51.18	59.29
CELEBA	ACSA	74.65	48.19	51.44	58.46	61.53
	F1	74.02	42.19	47.28	57.91	62.08
	GM	83.67	56.39	60.73	65.39	72.95

samples) for visual evaluation of the proposed framework. Figure 4, 5, 6, 7 and 8 presents 5 randomly generated samples per minority class of MNIST, Fashion-MNIST, SVHN, CIFAR-10 and CELEBA datasets respectively.

A.5 Model Architecture

For all image datasets (except ImageNet-100), the encoder, E_φ and the decoder, D_θ architectures are adapted from prior works (Chen *et al.*, 2016; Lucic *et al.*, 2018; Dai & Wipf, 2019; Mondal *et al.*, 2021). The architecture of the encoder and the decoder networks are same irrespective of the dataset chosen, as presented in Table 16. However, number of neurons in the 2^{nd} fully connected layer of decoder varies based on the dimension of the images. For ImageNet-100, the architecture of the encoder network is based on DenseNet201 (Huang *et al.*, 2017). The architectures of the mixer, M_ζ, the critic, C_ψ are fixed across all image datasets as mentioned in Table 17. Table 18 and Table 19 presents the model architectures used for tabular datasets.

Notation wise, CONV$_{n,k,s}$ denotes a convolutional layer with n kernels of size k and stride size s. TCONV$_{n,k,s}$ denotes a transpose convolutional layer with n kernels of size k and stride size s. FC$_n$ denotes a Fully Connected layer with n neurons. BN denotes a batch normalization layer. lReLU denotes Leaky Rectified Linear Unit activation. ReLU denotes Rectified Linear Unit activation. $Dropout(r)$ denotes a Dropout layer with dropout rate = r.

A.6 Training Details

In this section, we give detailed information about the training regime and hyperparameters which were used in the experiments shown in the main paper. We have compared our proposed method with four baseline methods, DeepSMOTE (Dablain *et al.*, 2022), GAMO (Mullick *et al.*, 2019), BAGAN (Mariani *et al.*, 2018), and DGC (Wang *et al.*, 2020). The architecture used for all these algorithms had parameters comparable to the architecture described in Table 16 to ensure a fair comparison. All of these methods were evaluated on the six well-known image datasets mentioned in Section A.1.

In our proposed method, the experimental settings except the latent space dimension were the same for all datasets. For MNIST (Lecun, 2010) and FashionMNIST (Xiao *et al.*, 2017) datasets, the latent space dimension used was 64, whereas for the SVHN (Netzer *et al.*, 2011) dataset it was set to 128. In the case of CIFAR-10 (Krizhevsky, 2009), CelebA (Liu *et al.*, 2015), CIFAR-100 (Krizhevsky, 2009), and ImageNet-100 (Deng *et al.*, 2009) datasets, it was chosen to be 256, 256, 256 and 512 respectively. For the tabular datasets (OPTICAL DIGITS, ISOLET, WEBPAGE, PROTEIN HOMO), we use a latent space of dimensionality 8. To learn the encoder network, we used AdamW optimizer with a learning rate of 2×10^{-4} and a weight decay of 10^{-3}. For all the other components of our method, we used Adam optimizer with a learning rate of 2×10^{-4}, β_1= 0.5 and β_2= 0.9.

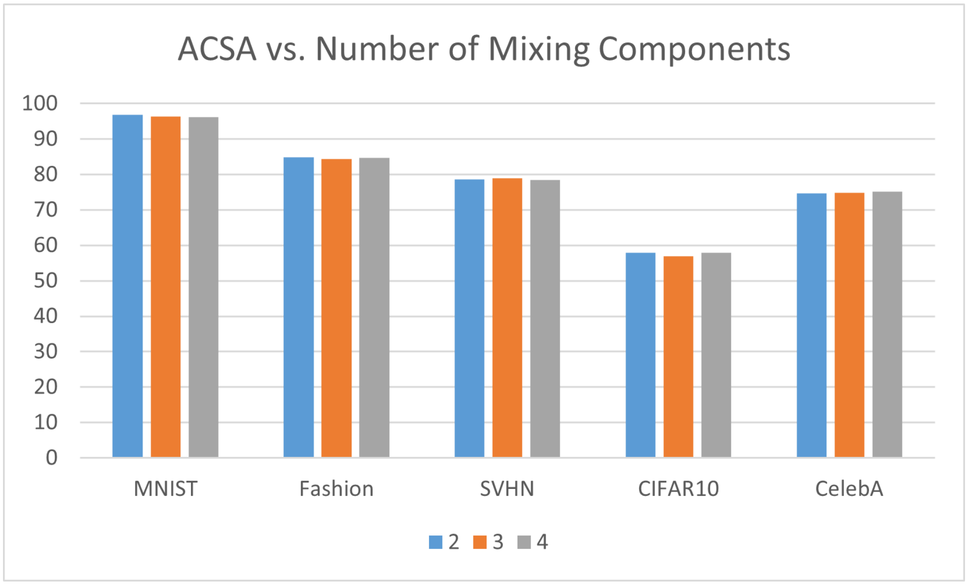


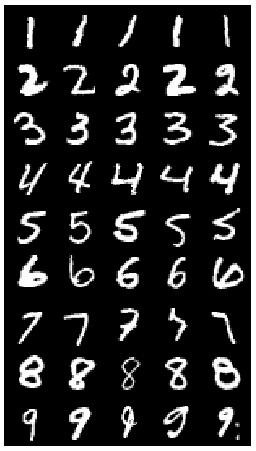
Figure 3: Ablation: Number of mixing components vs. Classification performance (ACSA) for different datasets.

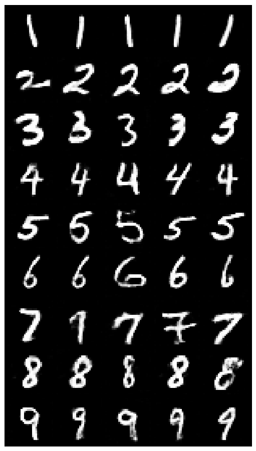
Table 16: Encoder and Decoder Architectures for Image Datasets

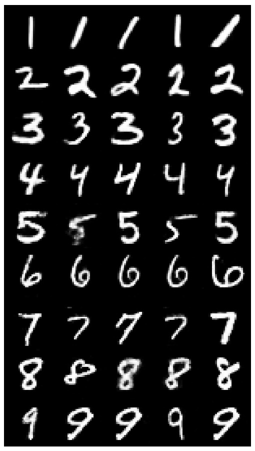
Encoder, (E_φ)	Decoder, (D_θ)
$\mathbf{x} \in \mathbb{R}^{h\times w\times c}$	$\mathbf{z} \in \mathbb{R}^m, \mathbf{y} \in \mathbb{R}^n$
$\to$ Conv$_{64,4,2}$ $\to$ BN $\to$ lReLU $\to$ Dropout(0.4)	$\to$ FC$_{1024}$ $\to$ BN $\to$ ReLU
$\to$ Conv$_{128,4,2}$ $\to$ BN $\to$ lReLU $\to$ Dropout(0.4)	$\to$ FC$_{\frac{h}{4}\times\frac{w}{4}\times 128}$ $\to$ BN $\to$ ReLU
$\to$ Flatten $\to$ FC$_{1024}$ $\to$ BN $\to$ lReLU $\to$ Dropout(0.4)	$\to$ Reshape$_{\frac{h}{4}\times\frac{w}{4}\times 128}$
$\to$ FC$_m$	$\to$ TCONV$_{128,4,2}$ $\to$ BN $\to$ ReLU
	$\to$ TCONV$_{64,4,2}$ $\to$ BN $\to$ ELU
	$\to$ CONV$_{c,3,1}$ $\to$ Sigmoid
$m = 64$ for MNIST, Fashion-MNIST; $m = 128$ for SVHN and $m = 256$ CIFAR10, CELEBA. $n = 10$ for MNIST, Fashion-MNIST, SVHN and CIFAR10, $n = 5$ for CELEBA.	

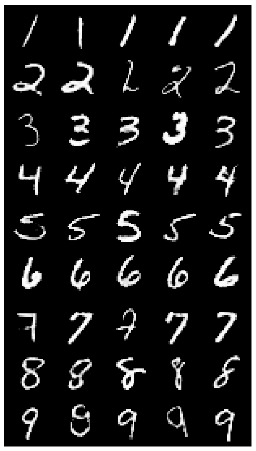
Table 17: Mixer and Critic Network Architectures for Image Datasets

Mixer, (M_ζ)	Critic, (C_ψ)
$\mathbf{z}_1 \in \mathbb{R}^m, \mathbf{z}_2 \in \mathbb{R}^m, \mathbf{y}_1 \in \mathbb{R}^n, \boldsymbol{n} \in \mathbb{R}^m \sim \mathcal{N}(0, I)$	$\boldsymbol{x} \in \mathbb{R}^{h\times w\times c}$
$\to$ FC$_{1024}$ $\to$ BN $\to$ lReLU $\to$ Dropout(0.4)	$\to$ Conv$_{64,4,2}$ $\to$ lReLU
$\to$ FC$_{1024}$ $\to$ BN $\to$ lReLU $\to$ Dropout(0.4)	$\to$ Conv$_{128,4,2}$ $\to$ lReLU
$\to$ FC$_{1024}$ $\to$ BN $\to$ lReLU $\to$ Dropout(0.4)	$\to$ Flatten $\to$ FC$_{1024}$ $\to$ lReLU
$\to$ FC$_{1024}$ $\to$ BN $\to$ lReLU $\to$ Dropout(0.4)	$\to$ FC$_1$
$\to$ FC$_2$ $\to$ Softmax	
$m = 64$ for MNIST, Fashion-MNIST; $m = 128$ for SVHN and $m = 256$ CIFAR10, CELEBA. $n = 10$ for MNIST, Fashion-MNIST, SVHN and CIFAR10, $n = 5$ for CELEBA.	


(a) Original (Lecun, 2010)


(b) Proposed Method


(c) GAMO2PIX (Mullick *et al.*, 2019)


(d) DeepSMOTE (Dablain *et al.*, 2022)

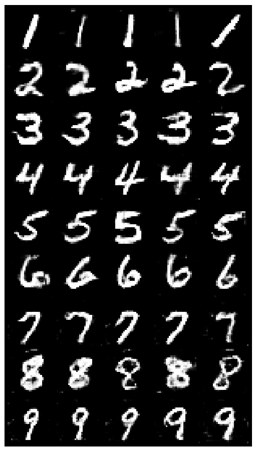
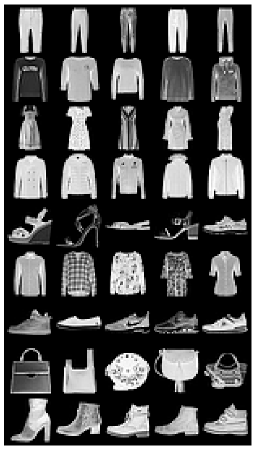
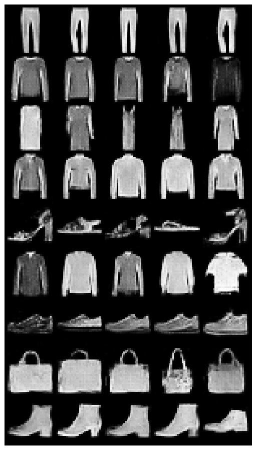
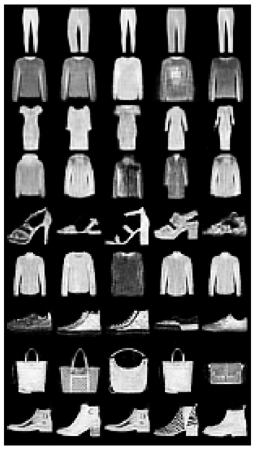
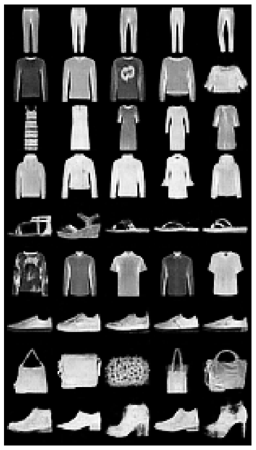

(e) BAGAN (Mariani *et al.*, 2018)

Figure 4: MNIST minority class images, with rows corresponding to digit classes. All the images are randomly chosen without any cherry picking.


(a) Original (Xiao *et al.*, 2017)


(b) Proposed Method


(c) GAMO2PIX (Mullick *et al.*, 2019)


(d) DeepSMOTE (Dablain *et al.*, 2022)

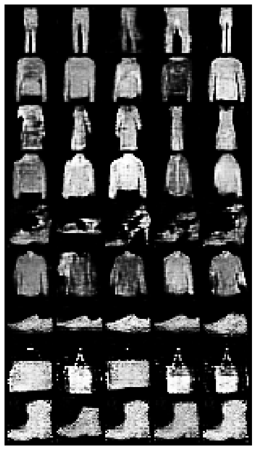
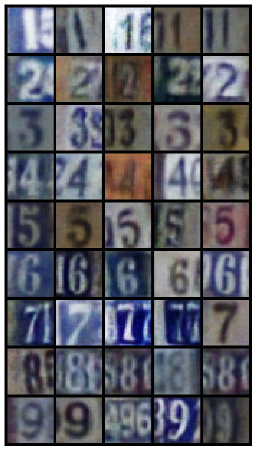

(e) BAGAN (Mariani *et al.*, 2018)

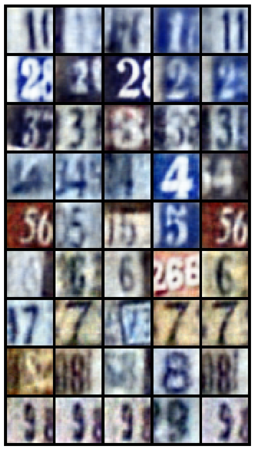
Figure 5: FashionMNIST (Xiao *et al.*, 2017) minority class images: trouser / pullover / dress / coat / sandal / shirt / sneaker / bag / ankle boot. All the images are randomly chosen without any cherry picking.

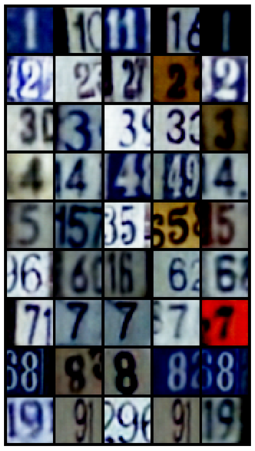
Table 18: Encoder and Decoder Architectures for Tabular Datasets

Encoder, (E_φ)	Decoder, (D_θ)
$\mathbf{x} \in \mathbb{R}^l$	$\mathbf{z} \in \mathbb{R}^m, \mathbf{y} \in \mathbb{R}^n$
$\rightarrow$ FC$_{512}$ $\rightarrow$ BN $\rightarrow$ lReLU $\rightarrow$ Dropout(0.4)	$\rightarrow$ FC$_{512}$ $\rightarrow$ BN $\rightarrow$ ReLU
$\rightarrow$ FC$_{256}$ $\rightarrow$ BN $\rightarrow$ lReLU $\rightarrow$ Dropout(0.4)	$\rightarrow$ FC$_{256}$ $\rightarrow$ BN $\rightarrow$ ReLU
$\rightarrow$ FC$_m$	$\rightarrow$ FC$_l$
$m = 8$	
$l = 64, 617, 300, 74$ for OPTICAL DIGITS, ISOLATE, WEBPAGE and PROTEIN HOMO.	

(a) Original (Netzer *et al.*, 2011)


(b) Proposed Method


(c) GAMO2PIX (Mullick *et al.*, 2019)


(d) DeepSMOTE (Dablain *et al.*, 2022)

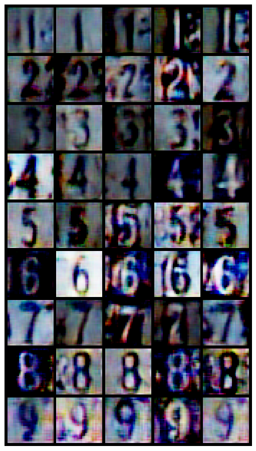
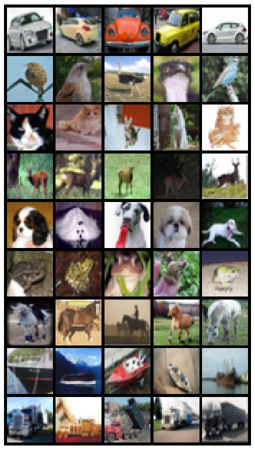
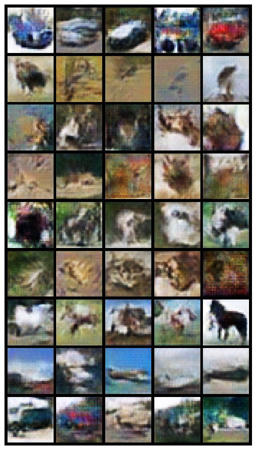
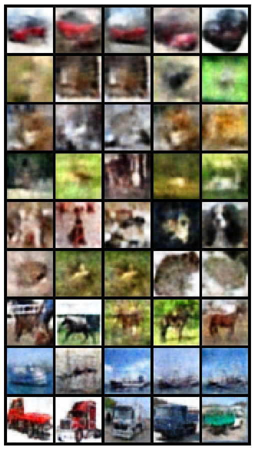
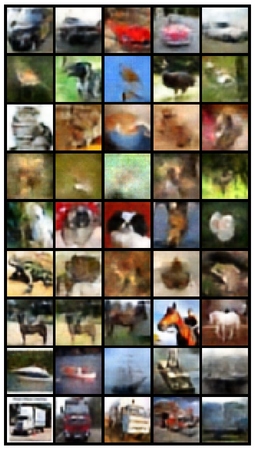

(e) BAGAN (Mariani *et al.*, 2018)

Figure 6: SVHN (Netzer *et al.*, 2011) minority class images, with rows corresponding to digit classes. All the images are randomly chosen without any cherry picking.


(a) Original (Krizhevsky, 2009)


(b) Proposed Method


(c) GAMO2PIX (Mullick *et al.*, 2019)


(d) DeepSMOTE (Dablain *et al.*, 2022)

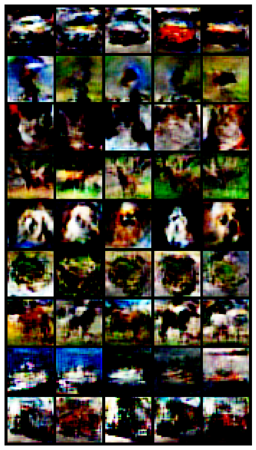
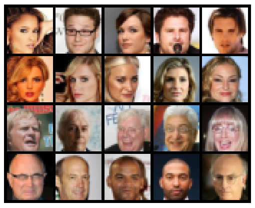
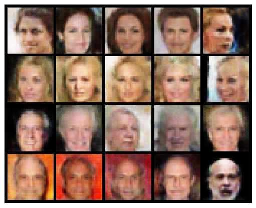
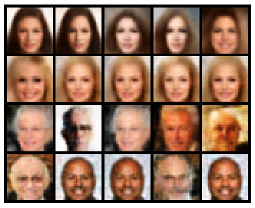
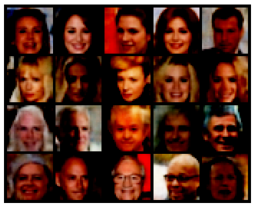

(e) BAGAN (Mariani *et al.*, 2018)

Figure 7: CIFAR-10 (Krizhevsky, 2009) minority class images: automobile / bird / cat / deer / dog / frog / horse / ship / truck. All the images are randomly chosen without any cherry picking.


(a) Original (Liu *et al.*, 2015)


(b) Proposed Method


(c) GAMO2PIX (Mullick *et al.*, 2019)


(d) DeepSMOTE (Dablain *et al.*, 2022)

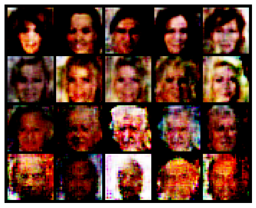

(e) BAGAN (Mariani *et al.*, 2018)

Figure 8: CelebA (Liu *et al.*, 2015) minority class images: brown hair / blond hair / gray hair / bald. All the images are randomly chosen without any cherry picking.

Table 19: Mixer and Critic Network Architectures for Tabular Datasets

Mixer, (M_ζ)	Critic, (C_ψ)
$\mathbf{z}_1 \in \mathbb{R}^m, \mathbf{z}_2 \in \mathbb{R}^m, \mathbf{y}_1 \in \mathbb{R}^n, \boldsymbol{n} \in \mathbb{R}^m \sim \mathcal{N}(0, I)$	$\boldsymbol{x} \in \mathbb{R}^l$
$\rightarrow$ FC$_{512}$ $\rightarrow$ BN $\rightarrow$ lReLU $\rightarrow$ Dropout(0.4)	$\rightarrow$ FC$_{512}$ $\rightarrow$ lReLU
$\rightarrow$ FC$_{512}$ $\rightarrow$ BN $\rightarrow$ lReLU $\rightarrow$ Dropout(0.4)	$\rightarrow$ FC$_{512}$ $\rightarrow$ lReLU
$\rightarrow$ FC$_{512}$ $\rightarrow$ BN $\rightarrow$ lReLU $\rightarrow$ Dropout(0.4)	$\rightarrow$ FC$_{512}$ $\rightarrow$ lReLU
$\rightarrow$ FC$_{512}$ $\rightarrow$ BN $\rightarrow$ lReLU $\rightarrow$ Dropout(0.4)	$\rightarrow$ FC$_1$
$\rightarrow$ FC$_2$ $\rightarrow$ Softmax	
$m = 8$	
$l = 64, 617, 300, 74$ for OPTICAL DIGITS, ISOLATE, WEBPAGE and PROTEIN HOMO.	