Learning to Re-weight Examples with Optimal Transport for Imbalanced Classification

Dandan Guo [1,2], Zhuo Li[3,4], Meixi Zheng[5], He Zhao [6], Mingyuan Zhou[7], Hongyuan Zha[1,8]
[1]School of Data Science, The Chinese University of Hong Kong, Shenzhen
[2] Institute of Robotics and Intelligent Manufacturing
[3]School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen
[4] Shenzhen Research Institute of Big Data
[5]Xidian University [6]CSIRO's Data61 [7]The University of Texas at Austin
[8]Shenzhen Institute of Artificial Intelligence and Robotics for Society
guodandan@cuhk.edu.cn 221019088@link.cuhk.edu.cn
meixizheng1110@163.com he.zhao@ieee.org
mingyuan.zhou@mccombs.utexas.edu zhahy@cuhk.edu.cn

Abstract

Imbalanced data pose challenges for deep learning based classification models. One of the most widely-used approaches for tackling imbalanced data is re-weighting, where training samples are associated with different weights in the loss function. Most of existing re-weighting approaches treat the example weights as the learnable parameter and optimize the weights on the meta set, entailing expensive bilevel optimization. In this paper, we propose a novel re-weighting method based on optimal transport (OT) from a distributional point of view. Specifically, we view the training set as an imbalanced distribution over its samples, which is transported by OT to a balanced distribution obtained from the meta set. The weights of the training samples are the probability mass of the imbalanced distribution and learned by minimizing the OT distance between the two distributions. Compared with existing methods, our proposed one disengages the dependence of the weight learning on the concerned classifier at each iteration. Experiments on image, text and point cloud datasets demonstrate that our proposed re-weighting method has excellent performance, achieving state-of-the-art results in many cases and providing a promising tool for addressing the imbalanced classification issue. The code has been made available at https://github.com/DandanGuo1993/reweight-imbalance-classification-with-OT.

1 Introduction

Deep neural networks (DNNs) have achieved remarkable success in various applications, which is undoubtedly inseparable from the high-quality large-scale datasets. Usually, the number of samples for each class in these datasets are manually selected resulting in balanced datasets. However, most real-world datasets are imbalanced, such as a few classes (a.k.a. head or majority class) occupy most of the data while most classes (a.k.a. tail or minority class) have a few samples. A model trained on the imbalanced training set but without considering such class imbalance would be significantly dominated by those majority classes, and thus underperform on a balanced test dataset. This can also be known as the long-tailed problem and exists in many domains, such as text classification [1, 2], object detection [3] and image classification [4–6].

There are rich research lines to solve the imbalance problem, including re-sampling [7–10], class-level or instance-level re-weighting [1, 2, 4, 11–18], meta-learning [4, 5, 15, 16, 19], two-stage methods

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

https://doi.org/10.52202/068431-1850

[4–6, 17] and post-hoc correction [20, 21]. Inspired by [2], re-weighting strategies can be roughly grouped into empirical re-weighting and automatic re-weighting. The former aims to design weights manually with the major insight that *the minority class example will be assigned a larger weight value than that of the majority class* [12–14]. However, manually setting weights can be less adaptive to different datasets [2]. The latter aims to assign adaptive weights to the examples through learning mechanisms [1, 2, 4, 15, 16]. As the representative automatic re-weighting method, L2RW [15] optimizes the weight vector as a learnable parameter with an unbiased meta set ($i.e.$, validation set). Although L2RW and its followers have received widespread attention, most of them may be limited to optimizing the weights by the classification loss on the meta set: The gradient of weights is usually coupled with the to-be-learned classifier at each training iteration. Since classifier is the major concern in imbalanced issue [6], the dependence of weights on classifier at training stage may lead to inaccurate learning of the weights.

This paper develops a novel automatic re-weighting method for imbalanced classification based on optimal transport (OT). As discussed by Jamal et al. [4], the major challenge for imbalanced classification is essentially the mismatch between the imbalanced training dataset (seen by a machine learning model) and the balanced test set (used to test the learned model). To this end, we aim to view the learning of the weight vector as the distribution approximation problem. We adopt the two-stage learning manner motivated by [6], where stage 1 and stage 2 focus on learning the feature extractor with the standard cross-entropy loss and the classifier with our proposed method, respectively. Specifically, we represent the imbalanced training set as a discrete empirical distribution P over all samples within it and view the to-be-learned weight vector $\boldsymbol{w}$ as its probability measure. Then we represent the balanced meta set as a discrete empirical distribution Q over all samples within it (in the same space with P), which has a uniform probability measure for being balanced. Therefore, the learning of a weight vector can be formulated as the process of learning the distribution P to be as close to the balanced distribution Q as possible, a process facilitated by leveraging the OT distance [22]. Notably, the cost function plays a paramount role when learning the transport plan for OT, where we use the features and ground-truth labels of samples to design it. Due to the flexibility of our method, we can also learn an explicit weight net directly from data like [16, 23] but with a different structure, optimized by OT loss instead of the classification loss on the meta set. Generally, at each training iteration at stage 2, we minimize the OT loss to learn the weight vector (or weight net) for the current mini-batch, which is further used to re-weight the training loss for optimizing the model. As we can see, the gradient of weights only relies on the OT loss and thus is independent of the classifier. More importantly, our proposed method is robust to the distribution Q. To save the memory consumption, we introduce the prototype-oriented OT loss by building a new distribution Q based on prototypes instead of samples (one prototype for each class). More importantly, our proposed method can achieve a reasonably good performance even if we randomly select a mini-batch from all prototypes to build Q, making our method applicable to datasets with a large number of classes.

We summarize our main contributions as follows: (1) We formulate the learning of weight vector or weight net as the distribution approximation problem by minimizing the statistical distance between to-be-learned distribution over samples from imbalanced training set and another balanced distribution over samples from the meta set. (2) We leverage the OT distance between the distributions to guide the learning of weight vector or weight net. (3) We apply our method to imbalanced classification tasks including image, text and point cloud. Experiments demonstrate that introducing the OT loss to learn the example weights can produce effective and efficient classification performance.

2 Related Work

Empirical Re-weighting A classic empirical re-weighting scheme is to provide the examples of each class with the same weight, such as inverse class frequency [11, 14]. It has been further improved by the class-balanced loss [13], which calculates the effective number of examples as class frequency. Focal Loss [12] uses the predicted probability to calculate higher weights for the hard examples and dynamically adjust the weights. LDAM-DRW [17] designs a label-distribution-aware loss function and adopts a deferred class-level re-weighting method ($i.e.$, inverse class frequency).

Automatic Re-weighting The automatic re-weighting methods learn the weights with learning mechanisms. L2RW [15] adopts a meta-learning manner to learn the example weights, which are optimized by the classification loss on the balanced meta set. Hu et al. [1] further improve L2RW

https://doi.org/10.52202/068431-1850

by iteratively optimizing weights instead of re-estimation at each iteration. Meta-weight-net [16] aims to learn an explicit weight net directly from data and optimize it by a meta-learning manner. Meta-class-weight [4] defines the weight for each example as the combination of class-level weight (estimated by Cui et al. [13]) and instance-level weight, optimized with a meta-learning approach similar to L2RW. Influence-balanced loss (IB) is proposed to [18] re-weight samples by the magnitude of the gradient. Recently, Liu et al. [2] propose to update the weights and model under a constraint. Our method belongs to automatic re-weighting group, and the idea of building an explicit weight net is similar to Shu et al. [16]. However, the major difference is that we bypass the classification loss on the meta set and use OT to learn the weights from the view of distribution approximation, disengaging the dependence of the weight learning on the concerned classifier at each iteration.

Meta Learning and Two-stage Learning Recently, researchers have proposed to tackle the imbalance issue with meta-learning, which can be applied to build a Balanced Meta-Softmax (BALMS)[19], learn weights [4, 15, 16] or transformed semantic directions for augmenting the minority classes in MetaSAug [5]. Two-stage methods, where the first stage and second stage focus on representation learning and classifier learning, respectively, have been proved effective for solving the imbalanced issue [5, 6, 16, 24]. BBN [25] unifies two stages with a specific cumulative learning strategy.

Optimal Transport Recently, OT has been used to solve the regression problem under the covariate shift [26], unsupervised domain adaption [27, 28], including sample-level, class-level or domain-level weight vector. Although they also adopt the re-weighting strategy and OT distance, they are distinct form ours in terms of task and technical detail. Also, the dynamic importance weighting which adopts MMD to re-weight samples for label-noise and class-prior-shift tasks [29] is also different from ours, where we provide a more flexible way for learning the weights of samples and disengage the dependence of the weight learning on the concerned classifier at each iteration. To the best of our knowledge, the works that solve imbalanced classification problem with OT are still very limited. An oversampling method via OT (OTOS) [30] aims to make synthetic samples follow a similar distribution to that of minority class samples. However, ours is a novel re-weighting method based on OT, without augmenting samples. Another recent work is Optimal Transport via Linear Mapping (OTLM) [21], which performs the post-hoc correction from the OT perspective and proposes a linear mapping to replace the original exact cost matrix in OT problem. Different from OTLM that belongs to the post-hoc correction group and aims to learn refined prediction matrix, ours falls into the training-aware group and aims to re-weight the training classification loss.

3 Background

Imbalanced Classification Consider a training set $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^{N}$, where (x, y) is the input and target pair, x_i the i-th sample, $y_i \in (0,1)^K$ the one-hot associated label vector over K classes, and N the number of the entire training data. Besides, consider a small balanced meta set $\mathcal{D}_{\text{meta}} = \{(x_j, y_j)\}_{j=1}^{M}$, where M is the amount of total samples and $M \ll N$. Denote the model parameterized with $\boldsymbol{\theta}$ as $f(x, \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is usually optimized by empirical risk minimization over the training set, *i.e.*, $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} \ell\left(y_i, f\left(x_i; \boldsymbol{\theta}\right)\right)$. For notational convenience, we denote $l_i^{\text{train}}(\boldsymbol{\theta}) = \ell\left(y_i, f\left(x_i; \boldsymbol{\theta}\right)\right)$ to represent the training loss function of pair (x_i, y_i). However, the model trained by this method will prefer the majority class if the training dataset is imbalanced.

Learning to Re-Weight Examples To solve the imbalanced issue, a kind of re-weighting methods is to treat the weights as the learnable parameter and learn a fair model to the minority and the majority classes by optimizing the weighted training loss. At each training iteration, the model is updated by

$$\boldsymbol{\theta}^*(\boldsymbol{w}) = \arg\min_{\boldsymbol{\theta}} \sum\nolimits_{i=1}^{N} w_i l_i^{\text{train}}(\theta), \tag{1}$$

where $\boldsymbol{w} = (w_1, \ldots, w_N)^T$ is the weight vector (usually with a simplex constraint) of all training examples. Then the optimal $\boldsymbol{w}$ is obtained by making the model parameter $\boldsymbol{\theta}^*(\boldsymbol{w})$ from Eq. (1) minimize the classification loss on a balanced meta set, formulated as

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \frac{1}{M} \sum\nolimits_{j=1}^{M} l_j^{\text{meta}}\left(\boldsymbol{\theta}^*(\boldsymbol{w})\right), \tag{2}$$

where l_j^{meta} is the loss function of pair (x_j, y_j) from meta set and the updated $\boldsymbol{w}^*$ is used to ameliorate the model. Generally, model $\boldsymbol{\theta}$ consists of two key components, feature extractor and classifier, where

https://doi.org/10.52202/068431-1850

the classifier has been proved to be the major concerning part in imbalanced issue [6]. However, the gradient of weights in Eq. (2) always depends on the to-be-concerned classifier at each training iteration, which may result in inaccurate learning of the weights. Most automatic re-weighting methods learn the weight vectors or weight-related parameters ($e.g.$, weight net) following this line; see more details from the previous works [4, 15, 16].

Optimal Transport Theory OT has been widely used to calculate the cost of transporting one probability measure to another in various machine learning problems, such as generative models [31], text analysis [32, 33], adversarial robustness [34], and meta learning [35, 36]. Among the rich theory of OT, this work presents a brief introduction to OT for discrete distributions; see Peyré and Cuturi [22] for more details. Consider $p = \sum_{i=1}^{n} a_i \delta_{x_i}$ and $q = \sum_{j=1}^{m} b_j \delta_{y_j}$ as two probability distributions, where x_i and y_j live in the arbitrary same space and δ is the Dirac function. Then, we can denote $\boldsymbol{a} \in \Delta^n$ and $\boldsymbol{b} \in \Delta^m$ as the probability simplex of $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively. The OT distance between p and q can be expressed as:

$$\mathrm{OT}(p, q) = \min_{\mathbf{T} \in \Pi(p,q)} \langle \mathbf{T}, \mathbf{C} \rangle, \tag{3}$$

where $\langle \cdot, \cdot \rangle$ is the Frobenius dot-product and $\mathbf{C} \in \mathbb{R}_{\geq 0}^{n \times m}$ is the transport cost matrix constructed by $C_{ij} = C(x_i, y_j)$. The transport probability matrix $\mathbf{T} \in \mathbb{R}_{>0}^{n \times m}$, which satisfies $\Pi(p, q) \coloneqq \{\mathbf{T} \mid \sum_{i=1}^{n} T_{ij} = b_j, \sum_{j=1}^{m} T_{ij} = a_i\}$, is learned by minimizing $\mathrm{OT}(p, q)$. Directly optimizing Eq. (3) often comes at the cost of heavy computational demands, and OT with entropic regularization is introduced to allow the optimization at small computational cost in sufficient smoothness [37].

4 Re-weighting Method with Optimal Transport

This work views a training set as a to-be-learned distribution, whose probability measure is set as learnable weight vector $\boldsymbol{w}$. We use OT distance to optimize $\boldsymbol{w}$ for re-weighting the training loss.

4.1 Main Objective

Given the imbalanced training set $\mathcal{D}_{\text{train}}$, we can represent it as an empirical distribution over N pairs, where each pair $(x_i, y_i)^{\text{train}}$ has the sample probability w_i ($i.e.$, the weight), defined as:

$$P(\boldsymbol{w}) = \sum\nolimits_{i=1}^{N} w_i \delta_{(x_i, y_i)^{\text{train}}}, \tag{4}$$

where $(x_i, y_i)^{\text{train}}$ is the i-th pair from the training set and the learnable weight vector $\boldsymbol{w}$ of all training examples means probability simplex of $\mathbb{R}^N$. Since the meta set $\mathcal{D}_{\text{meta}}$ is balanced for all classes and closely related with the training set, it is reasonable to assume that meta set has already achieved the balanced data distribution that the training set aims to approximate. For meta set, we thus can sample each pair from it with equal probability and present it with an empirical distribution Q:

$$Q = \sum\nolimits_{j=1}^{M} \frac{1}{M} \delta_{(x_j, y_j)^{\text{meta}}}, \tag{5}$$

where $(x_j, y_j)^{\text{meta}}$ is the j-th pair from the meta set. To learn $\boldsymbol{w}$, different from most automatic re-weighting methods, which minimize the classification loss on the meta set, we aim to enforce the to-be-learned distribution $P(w)$ to stay close to the balanced distribution Q. Here, we explore the re-weighting method by adopting the OT distance between $P(w)$ and Q:

$$\min_{\boldsymbol{w}} \mathrm{OT}(P(\boldsymbol{w}), Q) \overset{\text{def.}}{=} \min_{\boldsymbol{w}} \min_{\mathbf{T} \in \Pi(P(\boldsymbol{w}), Q)} \langle \mathbf{T}, \mathbf{C} \rangle, \tag{6}$$

where cost matrix $\mathbf{C} \in \mathbb{R}_{\geq 0}^{N \times M}$ is described below and transport probability matrix $\mathbf{T} \in \mathbb{R}_{>0}^{N \times M}$ should satisfy $\Pi(P(\boldsymbol{w}), Q) := \{\mathbf{T} \mid \sum_{i=1}^{N} T_{ij} = 1/M, \sum_{j=1}^{M} T_{ij} = w_i\}$.

4.2 Cost Function

For notation convenience, we reformulate the model as $f(x, \boldsymbol{\theta}) = f_2(f_1(x; \boldsymbol{\theta}_1); \boldsymbol{\theta}_2)$, where f_1 parameterized with $\boldsymbol{\theta}_1$ denotes the representation learning part before the classifier, and f_2 parameterized

with $\boldsymbol{\theta}_2$ denotes the classifier. Intuitively, the cost C_{ij} measures the distance between pair i in training set and pair j in meta set, which can be flexibly defined in different ways. We explore a few conceptually intuitive options of C_{ij}, although other reasonable choices can also be used.

Label-aware Cost As the first option, we can define C_{ij} with the ground-truth labels of two samples:

$$C_{ij} = \mathrm{d}^{\mathrm{Lab}}(y_i^{\text{train}}, y_j^{\text{meta}}), \tag{7}$$

where $\mathrm{d}^{\mathrm{Lab}}(\cdot,\cdot)$ also denotes a distance measure, and $y_i^{\text{train}}, y_j^{\text{meta}}$ are the ground-truth label vectors of the two samples, respectively. Intuitively, if we use the euclidean distance, then $\mathbf{C}$ is a $0-1$ matrix (we can transfer the non-zero constant to 1), *i.e.*, $C_{ij}=0$ if x_i^{train} and x_j^{meta} are from the same class, and $C_{ij}=1$ otherwise. Now the OT loss is influenced by neither feature extractor $\boldsymbol{\theta}_1$ nor classifier $\boldsymbol{\theta}_2$.

Feature-aware Cost Besides, we can define C_{ij} purely based on the features of samples:

$$C_{ij} = \mathrm{d}^{\mathrm{Fea}}(\boldsymbol{z}_i^{\text{train}}, \boldsymbol{z}_j^{\text{meta}}), \tag{8}$$

where $\boldsymbol{z}_i^{\text{train}} = f_1(x_i^{\text{train}}; \boldsymbol{\theta}_1) \in \mathbb{R}^E$ and $\boldsymbol{z}_j^{\text{meta}} = f_1(x_j^{\text{meta}}; \boldsymbol{\theta}_1) \in \mathbb{R}^E$ denote the E-dimensional representation of x_i^{train} and x_j^{meta}, respectively. $\mathrm{d}^{\mathrm{Fea}}(\cdot,\cdot)$ denotes any commonly used distance measure and we empirically find the cosine distance is a good choice. It is easy to see that if x_i^{train} and x_j^{meta}'s features are close, their cost is small. Here the OT loss is influenced by the feature extractor $\boldsymbol{\theta}_1$.

Combined Cost Finally, we can use both features and labels to define C_{ij}, denoted as

$$C_{ij} = \mathrm{d}^{\mathrm{Fea}}(\boldsymbol{z}_i^{\text{train}}, \boldsymbol{z}_j^{\text{meta}}) + \mathrm{d}^{\mathrm{Lab}}(y_i^{\text{train}}, y_j^{\text{meta}}). \tag{9}$$

Intuitively, C_{ij} will be small if two samples have the same label and similar features. Empirically, we find that using the $\mathrm{d}^{\mathrm{Fea}}=1-\mathrm{cosine}(\cdot,\cdot)$ and euclidean distance for $\mathrm{d}^{\mathrm{Lab}}$ gives better performance. Interestingly, given the feature-aware cost (8) or label-aware cost (7), the learned weight vector can be interpreted as the instance-level or class-level re-weighting method, respectively. The weight vector learned from the combined cost can be interpreted as the combination of class-level and instance-level weights, although no specialized design for two-component weights like previous [4]; see Fig. 1.

4.3 Learn the Weight Vector

Given the defined cost function, we adopt the entropy regularized OT loss [37] to learn the weight vector. We thus rewrite (6) as the following optimization problem:

$$\min_{\boldsymbol{w}} L_{\mathrm{OT}} = \left\langle \mathbf{C}, \mathbf{T}_\lambda^*(\boldsymbol{w}) \right\rangle, \text{ subject to } \mathbf{T}_\lambda^*(\boldsymbol{w}) = \underset{\mathbf{T}\in\Pi(P(\boldsymbol{w}),Q)}{\arg\min} \langle \mathbf{T}, \mathbf{C}\rangle - \lambda H(\mathbf{T}), \tag{10}$$

where $\lambda > 0$ is a hyper-parameter for the entropic constraint $H(\mathbf{T}) = -\sum_{ij} T_{ij} \ln T_{ij}$. Note that (10) provides us a new perspective to interpret the relationship between $\boldsymbol{w}$ and $\mathbf{T}$, where $\boldsymbol{w}$ is the parameter of the leader problem and $\mathbf{T}$ is the parameter of the follower problem, which is of the lower priority. Accordingly, when we minimize (10) with respect to $\boldsymbol{w}$ using gradient descent, we should differentiate through $\mathbf{T}$. Below we investigate the following two ways to optimize the weight vector.

Optimizing $\boldsymbol{w}$ directly Specifically, at each training iteration, we define $P(\boldsymbol{w})$ with current $\boldsymbol{w}$, use the Sinkhorn algorithm [37] to compute OT loss, then optimize $\boldsymbol{w}$ by $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} L_{\mathrm{OT}}$.

Amortizing the learning of $\boldsymbol{w}$ We also provide an alternative method by constructing an explicit weight net to output the example weights, whose structure can be designed flexibly. For example, we can build the following weight net and take the sample features as input:

$$\boldsymbol{w} = \mathrm{softmax}\left(\boldsymbol{s}\right), s^i = \boldsymbol{w}_{att} \tanh\left(\mathbf{W}_{vz} \boldsymbol{z}_i^{\text{train}}\right), \tag{11}$$

where s^i is the i-th element of $\boldsymbol{s} \in \mathbb{R}^N$, $\boldsymbol{w}_{att} \in \mathbb{R}^{1\times A}$ and $\mathbf{W}_{vz} \in \mathbb{R}^{A\times E}$ are the learned parameters (we omit the bias for convenience), denoted as $\boldsymbol{\Omega} = \{\boldsymbol{w}_{att}, \mathbf{W}_{vz}\}$. Denote $S(\boldsymbol{z}; \boldsymbol{\Omega})$ as the weight net parameterized by $\boldsymbol{\Omega}$, which can be optimized by $\boldsymbol{\Omega}^* = \arg\min_{\boldsymbol{\Omega}} L_{\mathrm{OT}}$.

5 Overall Algorithm and Implementations

To integrate our proposed method with deep learning frameworks, we adopt a stochastic setting, *i.e.*, a mini-batch setting at each iteration. Following [4, 5], we adopt two-stage learning, where

https://doi.org/10.52202/068431-1850

Algorithm 1 Workflow about our re-weighting method for optimizing $\boldsymbol{\theta}$ and $\boldsymbol{w}$.

Require: Datasets $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{meta}}$, initial model parameter $\boldsymbol{\theta}$ and weight vector, hyper-parameters $\{\alpha, \beta, \lambda\}$
for $t = 1, 2, ..., t_1$ **do**
  Sample a mini-batch B from the training set $\mathcal{D}_{\text{train}}$;
  Update $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_B$ where $\mathcal{L}_B = \frac{1}{|B|} \sum_{i \in B} \ell \left(y_i, f \left(x_i; \boldsymbol{\theta}^{(t)} \right) \right)$;
end for
for $t = t_1 + 1, ..., t_1 + t_2$ **do**
  Sample a mini-batch B from the training set $\mathcal{D}_{\text{train}}$;
  Step (a): Update $\hat{\boldsymbol{\theta}}^{(t+1)}(\boldsymbol{w}^{(t)}) \leftarrow \boldsymbol{\theta}^{(t)} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_B$ where $\mathcal{L}_B = \frac{1}{|B|} \sum_{i \in B} w_i^{(t)} \ell \left(y_i, f \left(x_i; \boldsymbol{\theta}^{(t)} \right) \right)$
  Use $\mathcal{D}_{\text{meta}}$ to build Q in (12) and B with $\boldsymbol{w}^t$ to build $P(\boldsymbol{w}^t)$ (4);
  Step (b): Compute $L_{\text{OT}} \left(\hat{\boldsymbol{\theta}}_1^{(t+1)}(\boldsymbol{w}^t), \boldsymbol{w}^{(t)} \right)$ with cost (9); Optimize $\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \beta \nabla_{\boldsymbol{w}} L_{\text{OT}} \left(\hat{\boldsymbol{\theta}}_1^{(t+1)}(\boldsymbol{w}^t), \boldsymbol{w}^{(t)} \right)$
  Step (c): Update $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_B$ where $\mathcal{L}_B = \frac{1}{|B|} \sum_{i \in B} w_i^{(t+1)} \ell \left(y_i, f \left(x_i; \boldsymbol{\theta}^{(t)} \right) \right)$
end for

stage 1 trains the model $f(\boldsymbol{\theta})$ by the standard cross-entropy loss on the imbalanced training set and stage 2 aims to learn the weight vector $\boldsymbol{w}$ and meanwhile continue to update the model $f(\boldsymbol{\theta})$. Generally, at stage 2, calculating the optimal $\boldsymbol{\theta}$ and $\boldsymbol{w}$ requires two nested loops of optimization, which is cost-expensive. Motivated by Hu et al. [1], we optimize $\boldsymbol{\theta}$ and $\boldsymbol{w}$ alternatively, corresponding to (1) and (10) respectively, where $\boldsymbol{w}$ is maintained and updated throughout the training, so that re-estimation from scratch can be avoided in each iteration. The implementation process of our proposed method with $\boldsymbol{w}$ optimized directly is shown in Algorithm 1, where the key steps are highlighted in Step (a), (b), and (c). Specifically, at each training iteration t, in Step (a), we have $\hat{\boldsymbol{\theta}}^{(t+1)}(\boldsymbol{w}^t) = \{\hat{\boldsymbol{\theta}}_1^{(t+1)}(\boldsymbol{w}^t), \hat{\boldsymbol{\theta}}_2^{(t+1)}(\boldsymbol{w}^t)\}$ and α is the step size for $\boldsymbol{\theta}$; in Step (b), as the cost function based on features is related with $\hat{\boldsymbol{\theta}}_1^{(t+1)}(\boldsymbol{w}^t)$, the OT loss relies on $\hat{\boldsymbol{\theta}}_1^{(t+1)}(\boldsymbol{w}^t)$, and β is the step size for $\boldsymbol{w}$; in Step (c), we ameliorate model parameters $\boldsymbol{\theta}^{(t+1)}$. We defer the learning of $\boldsymbol{\theta}$ and $\boldsymbol{\Omega}$ for the amortized learning of $\boldsymbol{w}$.

Discussion From Step (b), we find the gradient of $\boldsymbol{w}$ is unrelated to classifier $\boldsymbol{\theta}_2$ regardless of which cost function we choose. If we use the label-aware cost or freeze the feature extractor parameterized by $\boldsymbol{\theta}_1$, which is trained in the first stage, the OT loss in Step (b) can be further reduced as $L_{\text{OT}}(\boldsymbol{w}^t)$, where we only need Steps (b)-(c) at each iteration. This is different from most of automatic re-weighting methods, where the gradient of $\boldsymbol{w}$ is always related with the to-be-learned model $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}$ or classifier $\boldsymbol{\theta}_2$ (when freezing $\boldsymbol{\theta}_1$) for minimizing the classification loss on meta set.

Prototype-oriented OT loss (POT) Recall that we represent a balanced meta set with M samples as distribution Q in (5), where M/K is the number of data in each class and usually larger than 1. Computing the OT loss requires to learn a $B \times M$-dimensional transport matrix at each iteration. To improve the efficiency of algorithm, we average all samples from each class in the meta set to achieve its prototype and propose a new Q distribution over K prototypes:

$$Q = \sum\nolimits_{k=1}^{K} \frac{1}{K} \delta_{(\hat{x}_k, y_k)^{\text{meta}}}, \quad \hat{x}_k = \frac{K}{M} \sum\nolimits_{j=1}^{M/K} x_{kj}^{\text{meta}}, \tag{12}$$

where POT loss only needs a $B \times K$-dimensional transport matrix. Due to the robustness of our method to Q, when dealing with a large number of classes, we can randomly sample a mini-batch from K prototypes at each iteration to build Q.

6 Experiments

We conduct extensive experiments to validate the effectiveness of our proposed method on text, image, and point cloud imbalanced classification tasks. Notably, different from the imbalanced image

and point cloud classification, we find that optimizing the weight net is better than optimizing the weight vector directly in the text classification. Therefore, we optimize the weight vector for the image and point cloud cases and build a weight net for text case. Unless specified otherwise, we adopt the combined cost and set the hyper-parameter for the entropic constraint as $\lambda = 0.1$ and the maximum iteration number in the Sinkhorn algorithm as 200. We define the imbalance factor (IF) of a dataset as the data point amount ratio between the largest and smallest classes.

6.1 Experiments on Imbalanced Image Classification

Datasets and Baselines We evaluate our method on CIFAR-LT-10, CIFAR-LT-100, ImageNet-LT and Places-LT. We create ***CIFAR-LT-10*** (***CIFAR-LT-100***) from CIFAR-10 (CIFAR-100)[38] by downsampling samples per class with IF$\in \{200, 100, 50, 20\}$ [5, 13]. ***ImageNet-LT*** is built from the classic ImageNet with 1000 classes[39] and IF$=1280/5$ [5, 24]. ***Places-LT*** is created from Places-2 [40] with 365 classes and IF$=4980/5$ [4, 24]. We randomly select 10 training images per class as meta set [5]; see more details in Appendix B. We consider the following baselines: (1) **Cross-entropy (CE)**, the model trained on the imbalanced training set with CE loss. (2) **Empirical re-weighting methods**, like Focal loss [12], Class-balanced (CB) loss [13] and LDAM-DRW [17]. (3) **Automatic re-weighting methods**, including L2RW [15], IB [18], Meta-Weight-Net [16] and Meta-class-weight [4]. (4) **Meta-learning methods**, including MetaSAug [5] and above methods of [4, 15, 16, 19]. (5) **Two-stage methods**, such as OLTR [24], cRT [6], LWS [6], BBN [25] and methods of [4, 5].

Experimental details and results on CIFAR-LT For a fair comparison, we use ResNet-32 [41] as the backbone on CIFAR-LT-10 and CIFAR-LT-100. Following Li et al. [5], at stage 1, we use 200 epochs, set the learning rate α of $\boldsymbol{\theta}$ as 0.1, which is decayed by $1e^{-2}$ at the 160th and 180th epochs. At stage 2, we use 40 epochs, set α as $2e^{-5}$ and learning rate β of weights as $1e^{-3}$. We use the SGD optimizer with momentum 0.9, weight decay $5e^{-4}$ and set the batch size as 16. We list the recognition results of different methods on CIFAR-LT-10 and CIFAR-LT-100 with different imbalance factors in Table 1.We report the average result of 5 random experiments without standard deviation which is of small scale(e.g., 1e-2). We can see that our re-weighting method outperforms CE training by a large margin and performs better than the empirical or automatic re-weighting methods. Remarkably, our proposed method outperforms competing MetaSAug that conducts a meta semantic augmentation approach to learn appropriate class-wise covariance matrices when IF is 200, 100 and 50. Importantly, as the training data becomes more imbalanced, our method is more advantageous. Even though our proposed method is inferior to MetaSAug when the dataset is less imbalanced (IF$=20$), it can still achieve competing results and surpasses related re-weighting methods. This suggests that our proposed method can be used to enhance the imbalanced classification, without the requirement of designing complicated models or augmenting samples on purpose.

To more comprehensively understand our method, we provide a series of ablation studies on CIFAR-LT-100 with IF $= 200$ in Table 2. Firstly, to explore the impact of cost function, we use different cost functions for the OT loss. We can see that the combined cost performs better than label-aware cost and feature-aware cost, confirming the validity of combining features and labels to define cost. Besides, using either label-aware or feature-aware cost can still achieve acceptable performance, indicating the usefulness of OT loss in the imbalanced issue. Secondly, to explore the robustness of the meta distribution Q, we adopt three ways to build Q: (1) using prototypes defined in Eq. (12) (K samples) ; (2) using all samples defined in Eq. (5) ($10 * K$ samples) ; (3) randomly sampling one point from each class (K samples) in meta set. We find that prototype-based meta performs best, and the performance with random-sample meta or whole meta is still competitive, which demonstrates the robustness of our proposed method to the distribution Q and the benefit of using the prototypes to build Q. Third, we compare two ways for learning $\boldsymbol{w}$ in each iteration, where one is re-estimating $\boldsymbol{w}$ from scratch and another one is maintaining and updating $\boldsymbol{w}$ throughout the training ($i.e.$, iteratively optimizing weights). We find that iteratively optimizing performs better.

Since cost function is essential in optimizing the OT loss, we are interested in examining the learned weight vectors given by different cost functions. Here, we use CIFAR-LT-10, randomly choose $\{10, 9, ..., 1\}$ training samples from class $\{1, 2, ..., 10\}$ and obtain 55 samples, which are used to build distribution P. Besides, the 10 prototypes from meta set are used to build the distribution Q. Given the different cost functions, we show the learned weight vectors of 55 training samples in Fig. 1, which have very different properties. Specifically, the label-aware cost and feature-aware cost lead to class-level weights and sample-level weights, respectively. It is reasonable that label-aware cost

https://doi.org/10.52202/068431-1850

Table 1: Test top-1 errors (%) of ResNet-32 on CIFAR-LT-10 and CIFAR-LT-100 under different settings.

Datasets	**CIFAR-LT-10**				**CIFAR-LT-100**			
Imbalance Factor	200	100	50	20	200	100	50	20
CE loss(results from [5])	34.13	29.86	25.06	17.56	65.30	61.54	55.98	48.94
Focal loss [12] (results from [4])	34.71	29.62	23.29	17.24	64.38	61.59	55.68	48.05
CB, CE loss [13] (results from [5])	31.23	27.32	21.87	15.44	64.44	61.23	55.21	48.06
CB, Focal loss [13] (results from [4])	31.85	25.43	20.78	16.22	63.77	60.40	54.79	47.41
LDAM loss [17] (results from [5])	33.25	26.45	21.17	16.11	63.47	59.40	53.84	48.41
LDAM-DRW [17] (results from [5])	25.26	21.88	18.73	15.10	61.55	57.11	52.03	47.01
L2RW [15] (results from [16])	33.49	25.84	21.07	16.90	66.62	59.77	55.56	48.36
Meta-weight net [16]	32.80	26.43	20.90	15.55	63.38	58.39	54.34	46.96
Meta-class-weight with CE loss [4]	29.34	23.59	19.49	13.54	60.69	56.65	51.47	44.38
Meta-class-weight with focal loss [4]	25.57	21.10	17.12	13.90	60.66	55.30	49.92	44.27
Meta-class-weight with LDAM loss [4]	22.77	20.00	17.77	15.63	60.47	55.92	50.84	47.62
MetaSAug with CE loss [5]	23.11	19.46	15.97	12.36	60.06	53.13	48.10	**42.15**
IB [18]	27.85	23.47	18.34	14.59	60.34	54.61	51.07	46.43
IB+CB [18]	30.04	24.03	17.91	14.73	60.31	54.73	51.20	46.58
IB + Focal loss [18]	25.88	22.03	17.62	14.32	59.61	55.04	51.08	45.47
MetaSAug with focal loss [5]	22.73	19.36	15.96	12.84	59.78	54.11	48.38	42.41
MetaSAug with LDAM loss [5]	22.65	19.34	15.66	**11.90**	56.91	51.99	47.73	42.47
BBN [25]	-	20.18	17.82	-	-	57.44	52.98	-
Our method (Weight Vector)	**21.54**	**18.13**	**15.54**	12.50	**54.97**	**51.46**	**47.50**	42.85

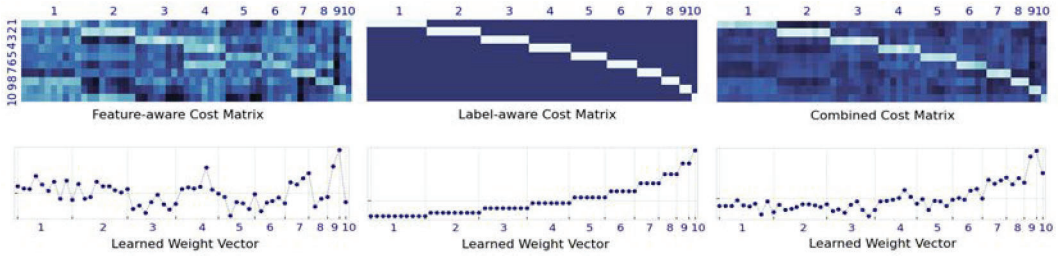

Figure 1: Learned weight vectors (bottom) given different cost functions (top) on CIFAR-LT-10, where x-axis denotes the 55 samples from the current mini-batch and where we only mark their labels for clarity.

only decides whether the two samples (from the meta set and training set) belong to the same class, resulting in class-level measure. However, feature-aware cost measures the distance between samples from the sample-level, where each sample has its own feature. More interestingly, the learned weights with the combined cost own the characteristics of class-level and sample-level weights simultaneously, where example weights of different classes are far away and example weights of the same class are close. Coincidentally, using the combined cost to define the OT loss can reach the same goal of [4], which explicitly considers class-level and sample-level weight. Besides, we find that the learned example weights of the minority class are usually more prominent than those of the majority classes.

Table 2: Ablation study on CIFAR-LT-100 with IF $= 200$, where $\boldsymbol{w}$ is maintained and updated throughout the training except the last row.

Method	**Top-1 errors**
Label +Prototype	55.06
Feature +Prototype	55.04
Combined +Prototype	**54.97**
Combined+ Whole	54.98
Combined +Random sample	55.03
Combined +Prototype+scratch	55.07

Table 3: Test top-1 errors(%) of ResNet-152 on Places-LT.

Method	**Places-LT**
CE	69.3
Focal loss [12] (from [24])	65.4
OLTR [24]	64.1
cRT [6]	63.3
LWS [6]	62.4
Mets-class-weight, CE[4]	62.9
Meta-softmax [19]	61.3
DisAlign[42]	60.7
L2RW + RANDOM[15]	67.77
Our method	**60.32±0.02**

Table 4: Test top-1 errors(%) of ResNet-50 on ImageNet-LT. $*$ indicates results from [5].

Method	**ImageNet-LT**
CE	61.12
CB, CE*[13]	59.15
OLTR*[24]	59.64
LDAM* [17]	58.14
LDAM-DRW* [17]	54.26
Mets-class-weight, CE*[4]	55.08
MetaSAug, CE [5]	52.61
Our method+Reduced Prototype	52.41
Our method	**52.36±0.01**

To verify whether our method ameliorates the performance on minority classes, we plot the confusion matrices of CE, MetaSAug, and ours on CIFAR-LT-10 with IF$=200$ in Fig. 2. As expected, although CE training can almost perfectly classify the samples in majority classes, it suffers severe performance degeneration in the minority classes. MetaSAug improves the accuracies of the minority classes, where is still a big gap between the performance on the minority classes and the majority classes. In

https://doi.org/10.52202/068431-1850

contrast, ours does not show a very clear preference for a certain class and outperforms the strong baseline on the overall performance, which is the goal of on an imbalanced classification task.

Experimental details and results on Places-LT and ImageNet-LT Following [6], we employ ResNet-152 pre-trained on the full ImageNet as the backbone on Places-LT. For stage 1, we set the initial learning rate as 0.01, which is decayed by $1e^{-1}$ every 10 epochs. In the stage 2 of our method, we only fine-tune the last fully-connected layer for training efficiency and set α as $1e^{-4}$ and β as $1e^{-3}$ within 50 epochs. The mini-batch size is 32 and the optimizer is SGD with momentum 0.9 and weight decay $5e^{-3}$. As shown in Table 3, our method outperforms all baselines. It further suggests that our method has excellent performance in the extreme imbalance setting with IF$=4980/5$. For a fair comparison, we implement our method on ImageNet-LT with the same experimental conditions of [5], from which we have taken the results of other comparison methods. We consider ResNet-50 [41] as the backbone on ImageNet-LT. In stage 1, we run 200 epochs and decay the learning rate by 0.1 at the 60th and 80th epochs. In stage 2, we implement our method for 50 epochs, set learning rate α as $2e^{-5}$ and β as $1e^{-2}$, and only fine-tune the last fully-connected layer for training efficiency. We use the SGD optimizer with momentum 0.9, weight decay $5e^{-4}$ and set the batch size as 128. The results on ImageNet-LT of different models reported in Table 4 indicate the effectiveness of our proposed method on ImageNet-LT when comparing with strong baseline MetaSAug. Besides, we further consider randomly sampling a mini-batch of size 100 from all prototypes at each iteration to build Q, whose performance is comparable to the Q from all prototypes. Thus, with a stochastic setting for Q, our proposed method can be used to the imbalanced training set with a large number of classes. We defer the time computational complexity, additional quantitative results and qualitative results on different image datasets to Appendix B.

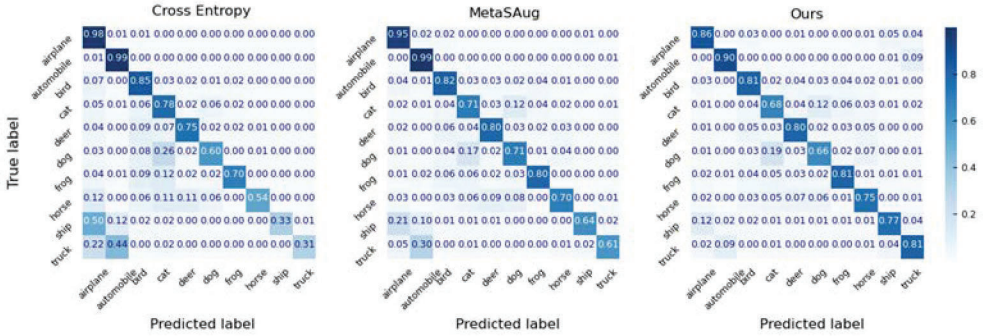

Figure 2: Confusion matrices of the cross-entropy training, MetaSAug and ours on CIFAR-LT-10 with the imbalance factor 200. We rank classes by the frequency, $i.e.$, frequent (left) and rare (right).

6.2 Experiments on Imbalanced Text Classification

Datasets and settings Following [1, 2], we adopt the popular SST-2 for 2-class and SST-5 for 5-class sentence sentiment [43]. For a fair comparison, we use the same imbalanced datasets and settings with [2]. Specifically, we set class 1 as the minority class and the rest as the majority classes, where the number of examples in the majority class is fixed as 1000 (SST-2) and 500 (SST-5) and we achieve different imbalance settings by varying the number of examples in the minority class. Besides, the number of samples in the meta set is 10 for each class. We use the BERT (base, uncased) model [44] as feature extractor and a simple 3-layer fully-connected network (FCN) with the structure in Appendix C as classifier. To make subsequent experiments on strong models, following [2], we use an additional balanced training set (500 samples in each class) to fine-tune the BERT model, which is randomly selected from the remaining examples in each dataset except the imbalanced training set, meta set and to-be evaluated test set. Based on the fine-tuned BERT, we adopt the two-stage manner for the imbalanced text datasets, where we train the BERT + FCN in the first stage with CE loss and train the FCN with our proposed method by freezing the BERT in the second stage. The settings of the training process are deferred to Appendix C.

Baselines We consider the following methods: (1) **vanilla BERT**, the vanilla pretrained language model. (2) **Fine-tuned BERT** , where the pretrained BERT is fine-tuned on an additional balanced training set. (3) **Fine-tuned BERT + CE**, the fine-tuned BERT model followed by the FCN which

https://doi.org/10.52202/068431-1850

is further trained by the CE loss on the imbalanced training set following [1, 2]. (4) **Automatic re-weighting methods**, including the method of Hu et al. [1] and constraint-based re-weighting [2]. Since few works consider imbalanced text classification, we further consider (5) **Empirical re-weighting methods**, including re-weighting with inverse class frequency ($i.e.$, Proportion) [11, 14]) and LDAM-DRW [17] and (6) **Logit adjustment** [20] using their official codes and settings[1] [2]. We repeat all experiments 10 times and report the mean and standard deviation.

Experimental details and results on SST-2 and SST-5 We report the text classification results of compared methods under different imbalance factors in Table 5. We find that our proposed method outperforms all competing methods in all imbalance factor settings, which demonstrates the effectiveness of our proposed method. Although all methods could achieve acceptable performance in a slight imbalance, the performance of three baselines (Vanilla BERT, Fine-Tuned BERT and Fine-Tuned BERT+CE) drop dramatically, indicating the importance of proposing specialized methods for handling imbalanced training datasets. Logit adjustment (post-hoc correction), is very competitive to ours on SST-2, which, however, only produces similar results to the three above-mentioned baselines on SST-5. In contrast, ours is robust to not only the imbalance factors but also the number of classes, where the results are consistent with the image case. We provide more results in Appendix C.4. In addition to 1D text and 2D image, we further investigate the robustness of our method on 3D point cloud data, where we use the popular ModelNet10 [45] and defer the experiments to Appendix D.

Table 5: Comparison of different models on SST-2 and SST-5. † indicates results reported in [2].

Method	**SST-2**				**SST-5**		
Imbalance Factor	1000 : 100	1000 : 50	1000 : 20	1000 : 10	500 : 75	500 : 60	500 : 50
Vanilla BERT	74.91±4.62	53.26±5.70	50.54±1.40	49.84±0.02	36.99±0.46	36.75±0.43	36.46±0.46
Fine-Tuned BERT	81.64±3.79	75.53±1.90	65.23±3.91	60.61±5.00	43.76±0.77	43.25±0.73	42.70±0.52
Fine-Tuned BERT+CE	78.25±2.24	57.18±1.88	55.00±1.23	50.17±1.34	43.71±0.98	44.06±1.11	36.46±0.50
Proportion (reported by us)	79.15±1.34	76.84±11.3	73.61±1.17	69.52±14.4	42.78±0.82	42.36±1.06	41.60±1.21
LDAM-DRW [17](reported by us)	71.41±1.25	64.65±3.82	56.41±3.55	53.73±3.01	43.20±0.64	40.90±1.66	40.81±1.46
Hu et al.'s † [1]	81.57±0.74	79.35±2.59	73.61±11.9	55.84±11.8	-	-	39.82±1.07
Hu et al.'s+Regularization † [1]	82.25±1.16	-	79.53±1.64	66.68±14.0	-	-	40.14±0.39
Constraint-based re-weighting † [2]	82.58±0.98	-	81.14±1.25	80.62±0.93	-	-	44.62±1.08
Logit Adjustment [20] (reported by us)	86.37±0.30	86.61±0.31	86.51±0.33	86.50±0.38	43.52±2.63	39.52±2.03	36.55±2.77
Our method (Weight Net)	**87.08±0.09**	**87.13±0.04**	**87.14±0.08**	**87.10±0.05**	**44.95±0.56**	**44.79±0.82**	**44.68±0.98**

7 Conclusion

This paper introduces a novel automatic re-weighting method for imbalance classification based on optimal transport (OT). This method presents the imbalanced training set as a to-be-learned distribution over its training examples, each of which is associated with a probability weight. Similarly, our method views another balanced meta set as a balanced distribution over the examples. By minimizing the OT distance between the two distributions in terms of the defined cost function, the learning of weight vector is formulated as a distribution approximation problem. Our proposed re-weighting method bypasses the commonly-used classification loss on the meta set and uses OT to learn the weights, disengaging the dependence of the weight learning on the concerned classifier at each iteration. This is an approach different from most of the existing re-weighting methods and may provide new thoughts for future work. Experimental results on a variety of imbalanced datasets of both images and texts validate the effectiveness and flexibility of our proposed method.

Acknowledgements. This work is partially supported by a grant from the Shenzhen Science and Technology Program (JCYJ20210324120011032) and Shenzhen Institute of Artificial Intelligence and Robotics for Society.

[1] https://github.com/kaidic/LDAM-DRW

[2] https://github.com/google-research/google-research/tree/master/logit_adjustment

References

[1] Zhiting Hu, Bowen Tan, Russ R Salakhutdinov, Tom M Mitchell, and Eric P Xing. Learning data manipulation for augmentation and weighting. *Advances in Neural Information Processing Systems*, 32, 2019.

[2] Yuqi Liu, Bin Cao, and Jing Fan. Improving the accuracy of learning example weights for imbalance classification. In *International Conference on Learning Representations*, 2022.

[3] Kemal Oksuz, Baris Can Cam, Sinan Kalkan, and Emre Akbas. Imbalance problems in object detection: A review. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(10):3388–3415, 2021. doi: 10.1109/TPAMI.2020.2981890. URL https://doi.org/10.1109/TPAMI.2020.2981890.

[4] Muhammad Abdullah Jamal, Matthew Brown, Ming-Hsuan Yang, Liqiang Wang, and Boqing Gong. Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7610–7619, 2020.

[5] Shuang Li, Kaixiong Gong, Chi Harold Liu, Yulin Wang, Feng Qiao, and Xinjing Cheng. Metasaug: Meta semantic augmentation for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5212–5221, 2021.

[6] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=r1gRTCVFvB.

[7] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.*, 16:321–357, 2002. doi: 10.1613/jair.953. URL https://doi.org/10.1613/jair.953.

[8] Hui Han, Wenyuan Wang, and Binghuan Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. In De-Shuang Huang, Xiao-Ping (Steven) Zhang, and Guang-Bin Huang, editors, *Advances in Intelligent Computing, International Conference on Intelligent Computing, ICIC 2005, Hefei, China, August 23-26, 2005, Proceedings, Part I*, volume 3644 of *Lecture Notes in Computer Science*, pages 878–887. Springer, 2005. doi: 10.1007/11538059_91. URL https://doi.org/10.1007/11538059_91.

[9] C. Drummond and R. C. Holte. C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. *proc of the icml workshop on learning from imbalanced datasets ii*, 2003.

[10] R. Barandela, R. M. Valdovinos, JS Sánchez, and F. J. Ferri. The imbalanced training sample problem: Under or over sampling? In *Structural, Syntactic, and Statistical Pattern Recognition, Joint IAPR International Workshops, SSPR 2004 and SPR 2004, Lisbon, Portugal, August 18-20, 2004 Proceedings*, 2004.

[11] C. Huang, Y. Li, C. C. Loy, and X. Tang. Learning deep representation for imbalanced classification. In *Computer Vision Pattern Recognition*, pages 5375–5384, 2016.

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

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

[14] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Learning to model the tail. *Advances in Neural Information Processing Systems*, 30, 2017.

[15] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pages 4334–4343. PMLR, 2018.

https://doi.org/10.52202/068431-1850

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

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

[18] Seulki Park, Jongin Lim, Younghan Jeon, and Jin Young Choi. Influence-balanced loss for imbalanced visual classification. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 715–724. IEEE, 2021. doi: 10.1109/ICCV48922.2021.00077. URL https://doi.org/10.1109/ICCV48922.2021.00077.

[19] Jiawei Ren, Cunjun Yu, Xiao Ma, Haiyu Zhao, Shuai Yi, et al. Balanced meta-softmax for long-tailed visual recognition. *Advances in Neural Information Processing Systems*, 33:4175–4186, 2020.

[20] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. *arXiv preprint arXiv:2007.07314*, 2020.

[21] Hanyu Peng, Mingming Sun, and Ping Li. Optimal transport for long-tailed recognition with learnable cost matrix. In *International Conference on Learning Representations*, 2022.

[22] Gabriel Peyré and Marco Cuturi. Computational optimal transport. *Found. Trends Mach. Learn.*, 11(5-6):355–607, 2019.

[23] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International conference on machine learning*, pages 2304–2313. PMLR, 2018.

[24] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2537–2546, 2019.

[25] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9719–9728, 2020.

[26] Julien Reygner and Adrien Touboul. Reweighting samples under covariate shift using a wasserstein distance criterion. *Electronic Journal of Statistics*, 16(1):3278–3314, 2022.

[27] Alain Rakotomamonjy, Rémi Flamary, Gilles Gasso, M El Alaya, Maxime Berar, and Nicolas Courty. Optimal transport for conditional domain matching and label shift. *Machine Learning*, 111(5):1651–1670, 2022.

[28] Rosanna Turrisi, Rémi Flamary, Alain Rakotomamonjy, and Massimiliano Pontil. Multi-source domain adaptation via weighted joint distributions optimal transport. *arXiv preprint arXiv:2006.12938*, 2020.

[29] Tongtong Fang, Nan Lu, Gang Niu, and Masashi Sugiyama. Rethinking importance weighting for deep learning under distribution shift. *Advances in Neural Information Processing Systems*, 33:11996–12007, 2020.

[30] Yuguang Yan, Mingkui Tan, Yanwu Xu, Jiezhang Cao, Michael K. Ng, Huaqing Min, and Qingyao Wu. Oversampling for imbalanced data via optimal transport. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 5605–5612. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33015605. URL https://doi.org/10.1609/aaai.v33i01.33015605.

[31] Viet Huynh, Dinh Q Phung, and He Zhao. Optimal transport for deep generative models: State of the art and research challenges. In *IJCAI*, pages 4450–4457, 2021.

[32] He Zhao, Dinh Phung, Viet Huynh, Trung Le, and Wray Buntine. Neural topic model via optimal transport. In *International Conference on Learning Representations*, 2021.

[33] Dongsheng Wang, Dandan Guo, He Zhao, Huangjie Zheng, Korawat Tanwisuth, Bo Chen, and Mingyuan Zhou. Representing mixtures of word embeddings with mixtures of topic embeddings. In *International Conference on Learning Representations*, 2022.

[34] Anh Tuan Bui, Trung Le, Quan Hung Tran, He Zhao, and Dinh Phung. A unified Wasserstein distributional robustness framework for adversarial training. In *International Conference on Learning Representations*, 2022.

[35] Dandan Guo, Tian Long, He Zhao, Mingyuan Zhou, and Hongyuan Zha. Adaptive distribution calibration for few-shot learning with hierarchical optimal transport. *https://arxiv.org/abs/2210.04144*, 2022.

[36] Dandan Guo, Long Tian, Minghe Zhang, Mingyuan Zhou, and Hongyuan Zha. Learning prototype-oriented set representations for meta-learning. In *International Conference on Learning Representations*, 2022.

[37] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26:2292–2300, 2013.

[38] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[39] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[40] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

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

[42] Songyang Zhang, Zeming Li, Shipeng Yan, Xuming He, and Jian Sun. Distribution alignment: A unified framework for long-tail visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2361–2370, 2021.

[43] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[44] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[45] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[46] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

 https://doi.org/10.52202/068431-1850

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 6.1. When handling imbalanced datasets of image classification, our proposed method is inferior to some competitive baseline method when the dataset is less imbalanced.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Appendix D.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] We've read the guidelines and make sure our submission adheres to the ethical standards.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Code will be submitted in the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 6.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] In Section 6.1, we report the average result of 5 random image experiments and omit standard deviation for the limited space, which is usually small (e.g., 1e-2). In Section 6.2, we repeat all text experiments 10 times and report the mean and standard deviation.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix B.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

https://doi.org/10.52202/068431-1850