Adaptive Cost-Sensitive Learning in Neural Networks for Misclassification Cost Problems

Ohad Volk*  
Faculty of Engineering  
Bar-Ilan University  
volkoha@biu.ac.il

Gonen Singer*  
Faculty of Engineering  
Bar-Ilan University  
gonen.singer@biu.ac.il

Abstract

We design a new adaptive learning algorithm for misclassification cost problems that attempt to reduce the cost of misclassified instances derived from the consequences of various errors. Our algorithm (adaptive cost sensitive learning – AdaCSL) adaptively adjusts the loss function such that the classifier bridges the difference between the class distributions between subgroups of samples in the training and test data sets with similar predicted probabilities (i.e., local training-test class distribution mismatch). We provide some theoretical performance guarantees on the proposed algorithm and present empirical evidence that a deep neural network used with the proposed AdaCSL algorithm yields better cost results on several binary classification data sets that have class-imbalanced and class-balanced distributions compared to other alternative approaches.
1 Introduction

Cost-sensitive learning is a domain that takes the misclassification costs resulting from various errors into account when training machine learning models. Research studies in this domain have designed sensitive classifiers such that insensitive models are modified to be biased toward classes with high misclassification costs. These types of machine learning models aim to achieve minimum misclassification costs (Domingos, 1999; Elkan, 2001; Ling et al., 2004). Usually, these approaches, which directly incorporate cost-sensitive capabilities into a learning algorithm, are specific to a group of classification methods and not generalizable.

Several research studies used a cost-sensitive learning approach to adjust insensitive deep neural networks by modifying misclassification costs based on the training data set, so that they can handle imbalanced data (Raj et al., 2016; Wang et al., 2016; Sze-To and Wong, 2017; Buda et al., 2018; Zhang et al., 2018; Dong et al., 2020). These studies usually used the misclassification costs to reflect the imbalance of the classes in the data, i.e., the ratio between misclassification costs reflects the ratio between classes in the training data set.

In many real-life problems, there are many situations in which both class imbalance and misclassification costs exist. For example, when classifying the severity of disease, the ratio between the cost of incorrectly classifying a severely ill patient as a healthy one and the cost of classifying a healthy patient as a severely ill one can be much higher than the ratio between healthy and severely ill patients in the data set. This paper addresses the above issues and proposes a cost-sensitive learning method for classification applications with misclassification error costs and class imbalances. The cost-learning parameters used to handle misclassification costs and class imbalance are considered differently in the learning process. In order to handle the class imbalance, the misclassification costs are used to modify the deep neural network toward the minority classes, and other learning parameters are used to reduce the misclassification costs. In addition, whereas all aforementioned studies, usually modified the misclassification costs based on the training data set to adjust deep neural networks, we propose a different approach, in which the loss function is adjusted adaptively, such that the classifier would be directed to reduce local high misclassification costs in the validation data set. In this setup, we develop an adaptive cost-sensitive learning (AdaCSL) algorithm that optimizes the loss function between consecutive learning epochs.

This paper makes four main contributions:

1. We propose a cost-sensitive learning method for classification applications with misclassification error costs and class-imbalanced problems.

2. We propose an adaptive algorithm that applies the cost-sensitive learning method, addressing local high misclassification costs in the validation data set by adjusting adaptively the loss function. The proposed algorithm can guide the design of a variety of different methods and loss functions.

3. The proposed algorithm applied in a convolutional neural network (CNN) and had been shown to outperform baseline and other cost-sensitive methods (over a test data set), based on four data sets reflecting different class imbalance ratios.

The remainder of this paper is organized as follows. In Section 2, we briefly discuss the related work on cost-sensitive learning approaches and the required adaptation to our problem. In Section 3, we introduce our proposed cost-sensitive learning method for handling local high misclassification costs in the validation data and describe the adaptive cost-sensitive learning (AdaCSL) algorithm. Section 4 presents our experiments and results when evaluating and benchmarking the proposed AdaCSL with a CNN against baseline and other cost-sensitive methods. Finally, Section 5 concludes this paper.

2 Preliminaries

We now review the cost-sensitive binary classification problem for individual instances in a training data set. Then we generalize the cost-sensitive approach for subgroups of samples with the same positive probability, aiming to identify local high misclassification costs.
2.1 The Cost-Sensitive Approach

First, we introduce notations used throughout the paper. Consider binary classification in which examples are taken from the instance space denoted by $X \in \mathcal{X} \subseteq \mathbb{R}^d$, where $d$ represents the number of features, and the output is denoted by $Y \in \{0, 1\}$, where 1 is the positive class and 0 is the negative class. A classification model $\mathcal{F} : \mathcal{X} \rightarrow [0, 1]$ maps an instance with input feature $X$ to a positive probability $P(Y = 1|X)$, or, briefly, $P(1|X)$, between 0 and 1, such that a higher probability expresses a stronger belief that an instance relates to the positive class. The negative probability, $P(Y = 0|X)$, or briefly, $P(0|X)$, expresses the belief that an instance relates to a negative class, and also ranges between 0 and 1. Given a threshold value $T$ and a positive probability $P(1|X)$, the instance is classified to the positive class if $P(1|X) > T$, and to the negative class otherwise.

In practical classification applications, inevitably, misclassification error costs are introduced. These costs derive from the consequences of misclassification errors. Cost-sensitive research takes these costs into account using three main approaches:

1. Data level approaches: Changing the distribution of classes, usually as pretraining approaches, using over- or under-sampling within the training data set (Elkan [2001]; Ling and Sheng [2008]), or weighting the training examples according to the misclassified samples, as post-training boosting approaches (Masnadi-Shirazi and Vasconcelos [2010]).

2. Algorithmic level approaches: Directly incorporating cost-sensitive capabilities into existing insensitive learning methods to be biased toward classes with high misclassification costs (Ling and Sheng [2008]; Lin et al. [2017]; Wu et al. [2018]; Aurelio et al. [2019]).

3. Decision level approaches: Using the original training data set and insensitive models, while adjusting the decision threshold $T'$ after modeling a classifier to minimize the misclassification costs (Hernández-Orallo et al. [2013]; Buda et al. [2018]).

Assume $C$ represents a misclassification cost matrix. The $(i, j)$ entry represents the misclassification cost of predicting instance $X$ as class $i$ when the actual class is $j$, denoted by $C_{(i, j)}$. The costs are constant across all instances in the data set for each type of classification error. Following (Ling and Sheng [2008]), who proved that any cost matrix could be converted into a matrix without incurring a cost to correct classifications such that the decision of classifying an example as positive will not be changed, we assume that $C_{(i, j)} = 0$, if $i = j$. The optimal prediction for an instance $X$ is the class $i$ that minimizes the total expected cost (risk), given by $R(X, i) = \sum_j P(j|X)C_{(i, j)}$, where $P(j|X)$ is the probability that an instance $X$ relates to class label $j$. Thus, the optimal prediction that minimizes the cost will be defined by $i = \arg\min_i R(X, i)$. It is shown (Elkan [2001]) that an optimal threshold (a decision level approach), which yields the minimum expected cost for a single instance, is defined by, $T = \frac{C_{(1, 0)}}{C_{(1, 1)} + C_{(1, 0)}}$. Insensitive learning algorithms try to achieve maximum accuracy while classifying instances according to the class with the majority probability and according to a fixed threshold of 0.5 for binary problems. Thus, it is proved (Elkan [2001]) that the number of negative class samples in the training data set should be multiplied by \( \left( \frac{1}{T'} \right) \left( \frac{1}{1-T'} \right) \) to make a threshold $T'$ corresponds to another threshold $T'$ (a data level approach), while $T' = 0.5$ generates a well-balanced data set.

Several research studies developed algorithmic level methods to convert specific insensitive classification models into sensitive ones, by incorporating cost-sensitive capabilities into the learning algorithm, thus attempting to reduce the total cost of misclassified instances. Most of these studies focus on decision trees models (Ting [2002]; Maimon and Rokach, 2014; Zhao and Li, 2017; Chaabane et al. 2019). In the context of deep neural networks, few papers tackle the data imbalance problem inherent in the learning process, while explicitly focusing on specific loss functions (Wang et al. 2016; Khan et al. 2017; Geng and Luo 2018; Wu et al. 2018; Aurelio et al. 2019). None of them, however, consider misclassification cost problems or problems with both misclassification costs and class imbalance. Furthermore, all existing studies, modified the deep neural networks based only on the training data, without considering misclassification costs from the validation data.

Our cost-sensitive learning hybrid method, in contrast, combines an algorithm-level approach and a decision-level approach. The algorithm level approach modifies the learners’ loss function to
alleviate their bias toward majority groups (negative class samples), by using the misclassification costs, while the decision level approach adaptively adjusts the loss function such that the classifier would be directed to reduce local high misclassification costs in the validation data set. The proposed method can be applied to any classifier with a loss function and is not restricted only to neural networks.

In Section 2.2, we divide a data set into subgroups of samples with the same predicted probabilities. We then prove that the threshold \( T \) holds for all instances in a training data set. This proof and the local subgroup concept constitute the basis for deriving the proposed adaptive cost-sensitive learning method for modifying deep neural networks so they address the performance degradation caused by local high misclassification costs.

### 2.2 Cost-Sensitive Approach For Subgroups Containing Samples With The Same Predicted Probabilities

Let \( D^t = (x_k, y_k), k = 1, ..., K \) be a training data set of labeled instances. Assume \( D^t_n, n = 1, ..., N \) presents a subgroup \( n \) of samples from the data set that are mapped to the same positive probability \( P(1|x_l) = P^t_n(1), \forall x_l \in D^t_n \) by a classification model.

**Theorem 1.** The optimal threshold for classifying instances from a data set is constant for all subgroups \( D^t_n, n = 1, ..., N \) and is equal to

\[
T = \frac{C_{(1, 0)}}{C_{(1, 0)} + C_{(0, 1)}},
\]

**Proof.** This proof follows the threshold proof for individual instances from [Elkan 2001]. Let \( |D^t_n| \) represent the number of instances in a subgroup. Assume an optimal threshold \( T^t_n \) for classifying samples in subgroup \( D^t_n \), which yields the minimum misclassification costs. Thus, if \( P^t_n(1) < T^t_n \), all instances belonging to subgroup \( D^t_n \) will be classified as negative class samples, and the cost of the errors emanating from the subgroup will be \( \sum_{x_l \in D^t_n} R(x_l, i = 0) = -P^t_n(1) \cdot |D^t_n| \cdot C_{(0, 1)} \).

If \( P^t_n(1) > T^t_n \), all instances belonging to subgroup \( D^t_n \) will be classified as positive class samples, and the cost will be \( \sum_{x_l \in D^t_n} R(x_l, i = 1) = (1 - P^t_n(1)) \cdot |D^t_n| \cdot C_{(1, 0)} \). The optimal threshold for making optimal decisions will be attained by comparing the two expressions and is equal to

\[
T^t_n = \frac{C_{(1, 0)}}{C_{(1, 0)} + C_{(0, 1)}}.
\]

Since the optimal threshold does not depend on subset \( D^t_n \) or the positive probability \( P^t_n(1) \), the optimal threshold for all instances in the training data set is equal, i.e., \( T = \frac{C_{(1, 0)}}{C_{(1, 0)} + C_{(0, 1)}} \).

Insensitive learning algorithms try to achieve maximum accuracy in the applied phase by minimizing the loss in the training phase. After the training phase, to evaluate the accuracy of the model’s prediction, the samples are classified according to the class with the greater probability and according to a fixed threshold of 0.5 for binary problems. In contrast, the decision-level approach for cost-sensitive problems, which seeks to decrease misclassification costs, classifies samples according to different values of a threshold calculated using the problem’s misclassification costs, as proved in Theorem 1. Thus, to achieve a specific and constant threshold \( T' \) which does not necessarily depend on the costs (a specific case, for example, can be \( T' = 0.5 \)), we follow [Elkan 2001] to find the expression in which we should multiply the number of negative samples in each subgroup. In Lemma 1, we prove that this expression is equal for all subgroups.

**Lemma 1.** When the optimal thresholds of subgroups in the training data set are \( T^t_n = T, \forall n = 1, ..., N \), to make a target probability threshold of a training data set correspond to a given probability threshold \( T' \), the number of negative examples in every subgroup \( D^t_n \) in the training data set should be multiplied by

\[
\left( \frac{C_{(1, 0)}}{C_{(0, 1)}} \right) \left( \frac{1 - T'}{T'} \right)
\]

**Proof idea.** This lemma is a generalization of Theorem 1 in [Elkan 2001] for subgroups of samples, and follow his proof. The main idea is that similar to Elkan, the number of negative class samples in a subgroup \( D^t_n \), should be multiplied by \( \left( \frac{T'}{1 - T^t_n} \right) \left( \frac{1 - T'}{T'} \right) \) to make a threshold
we formulate the mean loss over the training set considering the cost matrix $T_n^t$, assuming the number of negative instances in each subgroup is $\sum_{l=1}^{|D_n^t|} (1 - y_l)$, the new number of negative samples in the training data set will be $\sum_{n=1}^{N} \left( \frac{T_n^t}{1-T_n^t} \right) \left( \frac{1-T_n^t}{T_n^t} \right) \sum_{l=1}^{|D_n^t|} (1 - y_l)$. Following Theorem 1, with constant $T_n^t$ and assuming the same given threshold for all subgroups $T'$, we obtain $\left( \frac{T_n^t}{1-T_n^t} \right) \left( \frac{1-T_n^t}{T_n^t} \right) \sum_{n=1}^{N} \sum_{l=1}^{|D_n^t|} (1 - y_l)$, where $\sum_{n=1}^{N} \sum_{l=1}^{|D_n^t|} (1 - y_l)$ represents the number of negative class samples in the entire training data set. Using $T_n^t = T$ from Theorem 1, the number of negative class samples in every subgroup, or equivalently in the training data set, should be multiplied by $\left( \frac{C(1,0)}{C(0,1)} \right) \left( \frac{1-T'}{T'} \right)$.

3 An Adaptive Cost-Sensitive Learning Method

This section introduces a cost-sensitive learning method in neural networks using a modification of a loss function by utilizing both a decision threshold approach and an algorithmic level approach. The first expression, found in Section 3.1, modifies the loss function to alleviate their bias towards majority groups caused by the class imbalance problem, by using the misclassification costs. The second expression, found in Section 3.2, adjusts the loss function by an adaptive cost-sensitive learning (AdaCSL) algorithm presented in Section 3.3, aiming to address local high misclassification costs in the validation data set. We adopt the terminology and findings presented in Section 2 and apply them to subgroups of samples in the validation data set.

3.1 Cost-Sensitive Loss Function

A general cost-sensitive learning method is developed by modifying the loss function used in neural networks, aiming to reduce misclassification error costs. The proposed method is suitable for class-balanced and class-imbalanced problems with predefined misclassification costs. For this purpose, we formulate the mean loss over the training set considering the cost matrix $C$ as

$$E(\theta) = \frac{1}{K} \sum_{k=1}^{K} \mathcal{L}(y_k, \hat{y}_k^\theta, C) \tag{3}$$

where the predicted output of the $k^{th}$ sample $\hat{y}_k^\theta$ is parameterized by the weights and biases $\theta$ of the network, which is denoted by $\hat{y}_k$. Different from other research studies, here the cost matrix is constant and predefined. The loss function $\mathcal{L}(\cdot)$ can be any suitable surrogate loss. The objective of the learning process is to find the optimal $\theta^*$, which yields the minimum possible error, expressed by $\theta^* = \arg\min_{\theta} E(\theta)$. We discuss the Cross-Entropy loss (CE) for binary classification, which measures the distance from the actual class (0 or 1) to the predicted value, $\hat{y}_k = P(1|x_k)$, and can be expressed as $\mathcal{L}(y_k, \hat{y}_k) = -y_k \log(\hat{y}_k) - (1 - y_k) \log(1 - \hat{y}_k)$.

**Theorem 2.** When the optimal thresholds of subgroups of a training data set are equal $T_n^t = T$, $\forall n = 1, ..., N$, to make a target probability threshold $T$, calculated in eq. (1), corresponding to a given probability threshold $T'$, the CE loss should be

$$\mathcal{L}(y_k, \hat{y}_k, C) = -y_k \log(\hat{y}_k) - \left( \frac{C(1,0)}{C(0,1)} \right) \left( \frac{1-T'}{T'} \right) (1 - y_k) \log (1 - \hat{y}_k) \tag{4}$$

**Proof.** The total CE loss of a subgroup $D_n^t$, with the same positive class probability $\hat{y}_t = \hat{y}_n^t$, $\forall x_t \in D_n^t$ is $\sum_{l=1}^{|D_n^t|} \mathcal{L}(y_l, \hat{y}_l^t) = - \sum_{l=1}^{|D_n^t|} (y_l \log(\hat{y}_l^t) + (1 - y_l) \log(1 - \hat{y}_l^t))$. Since $\hat{y}_l^t$ is constant and does not depend on a sample $x_t$, we obtain, $\sum_{l=1}^{|D_n^t|} \mathcal{L}(y_l, \hat{y}_l^t) = - \log(\hat{y}_l^t) \sum_{l=1}^{D_n^t} y_l - \log(1 - \hat{y}_l^t) \sum_{l=1}^{D_n^t} (1 - y_l)$. The expression $\sum_{l=1}^{D_n^t} (1 - y_l)$ reflects the number of negative samples in subgroup $D_n^t$. The effect of changing the number of negative class samples in a subgroup as presented in Lemma 1 is equivalent to multiplying this expression in the CE loss function. Thus, to make a probability threshold $T_n^t = T$ of a subgroup, defined in eq. (1), correspond to a given probability threshold $T'$, the loss function
should be \[ \sum_{t=1}^{|D'|} L(y_t, \hat{y}_t) = \sum_{t=1}^{|D'|} (y_t \log(\hat{y}_t) + \left( \frac{C(1, 0)}{C(0, 1)} \right) \left( \frac{1-T'}{T'} \right) (1-y_t) \log(1-\hat{y}_t) \). \] Following Theorem 1, since the target threshold for all subgroups in the training data set is similar, the loss function is \[ L(y_k, \hat{y}_k) = -y_k \log(\hat{y}_k) - \left( \frac{C(1, 0)}{C(0, 1)} \right) \left( \frac{1-T'}{T'} \right) (1-y_k) \log(1-\hat{y}_k), \forall (x_k, y_k) \in D'. \]

### 3.2 Loss Function Adjustment For Addressing Local High Misclassification Costs

We propose to adjust the loss function between two consecutive epochs, in order to enable neural networks to reduce local high misclassification costs in the validation dataset, without changing the validation data set. Let \( D' = (x_r, y_r), r = 1, \ldots, R \) be a validation data set of labeled instances. Assume \( D'_m, m = 1, \ldots, M \) represent subgroups of samples among the validation data set that are mapped to the same positive probability \( P(1|x_j) = P'_m(1), \forall x_j \in D'_m \) by a classifier, denoted by \( P'_m \).

**Theorem 3.** To make an actual threshold of subgroup \( T'_m, m \), which yields minimum costs in the validation data set, correspond to a probability threshold \( T' \), the CE loss for the samples in subgroup \( (x_j, y_j) \in D'_m \) should be

\[
L(y_j, \hat{y}_j) = -y_j \log(\hat{y}_j) - \left( \frac{C(1, 0)}{C(0, 1)} \right) \left( \frac{1-T'}{T'} \right) (1-y_j) \log(1-\hat{y}_j) \quad (5)
\]

**Lemma 2.** The predicted class probability \( \hat{y}_k = P(1|x_k) \) of an instance \( k \) should be changed to \( \hat{y}'_k \) in response to a change in the prior probability of an instance in the data set from \( P = P(1) \) to \( P' = P'(1) \) such that

\[
\hat{y}'_k = P' \frac{\hat{y}_k - \hat{y}_k P}{P - \hat{y}_k + P' \hat{y}_k - PP'}
\]

**Proof.** Lemma 2 was proved in [Elkan, 2001], using Bayes’ rule. Given this lemma, we can now prove Theorem 3.

**Proof of Theorem 3.** In Theorem 2, we proved that the second expression should be multiplied by \( \frac{\hat{y}_k}{T'} \) to yield the minimum cost at \( T' \) in the training data set. According to Lemma 2, since the actual probability of being positive is different from the predicted positive probability \( P'_m \neq P'_m \), the actual threshold that yields the minimum cost is \( T'_m \neq T' \). Suppose that \( P'_m = \frac{P'_m}{P'_m} \) and \( P''_m = \frac{P''_m}{P''_m} \); thus, the number of actual negative samples should be multiplied by \( \frac{P''_m}{P'_m} \) to modify \( P'_m \) so that it corresponds to \( P''_m \). From Lemma 2, assuming that \( \hat{y}'_k = T'_m \) and \( \hat{y}_k = T' \), \( P''_m \) can be expressed as \( P''_m = \frac{T'_m P''_m}{P''_m - T'_m P''_m + T'_m P''_m} \) and thus the number of actual negative samples should be multiplied by \( \frac{T'_m P''_m}{P''_m - T'_m P''_m + T'_m P''_m} \)\( \frac{P''_m}{P'_m} \). By the first-order Taylor series expansion around the point \( T' \) of the function,

\[
\left( \frac{T'_m}{P''_m} \right) \left( \frac{1-T'}{T'} \right) \text{ yields } \left( \frac{T'_m}{P''_m} \right) \left( \frac{1-T'}{T'} \right) \approx e^{-\frac{T''(1-T')}{T'}}.
\]

Thus, the total number of negative class samples \( \sum_{j=1}^{|D'_m|} (1-y_j) \) in the loss of subgroup \( D'_m \) should be multiplied by

\[
\lambda = e^{-\frac{T''(1-T')}{T'}}. \quad (7)
\]

Similar to Theorem 2, since we use the same optimal threshold for all samples in the subgroup, each sample in the validation data set \( (x_j, y_j) \in D'_m \) should be multiplied by the same expression and the loss function should be \( L(y_j, \hat{y}_j, \lambda) = -y_j \log(\hat{y}_j) - \lambda \left( \frac{C(1, 0)}{C(0, 1)} \right) \left( \frac{1-T'}{T'} \right) (1-y_j) \log(1-\hat{y}_j) \).
Algorithm 1 Adaptive Cost-Sensitive Learning (AdaCSL)

**Input:** Training data set $D^t = (X^t, Y^t)$, validation data set $D^v = (X^v, Y^v)$, number of batches $B$, cost matrix $C$, model $F$, targeted threshold $T'$, set of potential thresholds $T$

**Output:** Learned parameters $\theta^*$

**Initialize:** $\theta$ arbitrarily, $i=1$, $\lambda_1^\psi = 1$

repeat
  for $b = 1$ to $B$ do
    $Y^t_b \leftarrow F(X^t_b; \theta)$  // Probability
    $\theta \leftarrow \theta - \alpha \nabla_{\theta} L(Y^t_b, \hat{Y}^t_b, \lambda_1^\psi)$
  end for
  $Y^v \leftarrow F(X^v; \theta)$  // Probability
  Divide $D^v$ into subgroups $D^v_m$, $m = 1, ..., M$
  for $m = 1$ to $M$ do
    $T^{v,a}_m \leftarrow \arg\min_{\tau \in T} Cost(\hat{Y}^v_m, Y^v_m, \tau, C)$
  end for
  $\lambda_{i+1}^\psi \leftarrow \lambda_i^\psi \cdot \frac{\sum_{m=1}^{M} |D^v_m| e^{-\left(\frac{T'^{-1}}{T'}\right)(1-y_r)}}{R}$ // Adaptive
  $i \leftarrow i + 1$
until $(\lambda_{i+1}^\psi - \lambda_i^\psi) < \epsilon$

as follows:

$$\lambda^\psi = \frac{\sum_{m=1}^{M} |D^v_m| e^{-\left(\frac{T'^{-1}}{T'}\right)(1-y_r)}}{R}. \quad (8)$$

Thus, the following loss function with an averaged adjustment $\forall (x_r, y_r) \in D^v$ is

$$L(y_r, \bar{y}_r, \lambda^\psi) = -y_r \log(\bar{y}_r) - \lambda^\psi \left(\frac{C(1, 0)}{C(0, 1)} \left(1 - T'\right) \left(1 - y_r\right) \log(1 - \bar{y}_r). \quad (9)$$

### 3.3 Adaptive Cost-Sensitive Learning Algorithm

Our goal is to develop an adaptive cost-sensitive learning (AdaCSL) algorithm for any neural network, aiming to achieve minimum misclassification costs. The algorithm learns the weight and bias parameters $\theta$ in the training process, while adjusting the loss function between each consecutive epoch, using the validation data set. In each epoch, we keep the loss function constant and use gradient descent with the backpropagation of eq. (3). To adjust the loss function, we use the neural networks with the current estimate of parameters $\theta$ and divide the validation data set $D^v = (x_r, y_r), r = 1, ..., R$ into $M$ subgroups $D^v_m, m = 1, ..., M$, each containing samples that are mapped to a similar positive probability $P^v_m$. For each subgroup we find the optimal thresholds $T^{v,a}_m, m = 1, ..., M$ among a threshold set $T$ yielding the minimum misclassification costs and calculate $\lambda$ from eq. (7). In the next training epoch, the loss function is updated using $\lambda^\psi$ from eq. (8). The adaptive process ends when the loss function is not significantly adjusted, as defined by $\epsilon$.

### 4 Experiments

In this section, the AdaCSL algorithm used in Convolutional Neural Network (CNN) is applied in two experiments with different objectives. The first experiment aims to illustrate the algorithm’s novelty in resolving local high misclassification costs in the validation data set by adjusting the loss function between two consecutive learning epochs. The objective of the second experiment is to evaluate the AdaCSL algorithm compared to other CNNs, some with other cost-sensitive approaches.
4.1 Illustration Of Addressing Local High Misclassification Costs By The AdaCSL Algorithm

4.1.1 Data sets, experimental setup, and method

We illustrate the AdaCSL algorithm’s novelty in directing deep neural network to reduce local high misclassification costs in the validation data set, based on the task of distinguishing between photos of wolves and Eskimo dogs (Ribeiro et al., 2016). We use a class-balanced training data set with 1,200 images and class-balanced validation data set with 400 images and identical misclassification costs \( C_{1,0} = C_{0,1} = 2 \) (i.e., local misclassification improvement is equivalent to local accuracy improvement). We use the PyTorch (Paszke et al., 2019) implementation of the ResNet18 (He et al., 2016) architecture for our experiment. Since the dataset is relatively small, we use a pre-trained version of ResNet on ImageNet (Deng et al., 2009) to reduce the initial feature extraction phase’s importance. We use the Stochastic Gradient Descent (SGD) algorithm to learn the parameters and regularize with weight decay. In order to illustrate the effect of the adjustment of the loss function using the AdaCSL algorithm, we use the following mechanism for two consecutive learning epochs. In Epoch 1, by using a standard Cross-Entropy (CE) as the loss function, we generated probability estimations for both training and validation data sets. We split the training samples into 10 subgroups \( D^m_m \), \( m = 1, \ldots, 10 \) and the test samples into 10 subgroups \( D^v_m \), \( m = 1, \ldots, 10 \), such that \( P(wol|x) \in [0.1 \cdot m - 0.1, 0.1] \), \( \forall x \in D_m = (D^m_m, D^v_m) \). For the following epoch 2, we build both CNN based on a standard CE (CNN w/o AdaCSL) and CNN with AdaCSL algorithm (CNN w AdaCSL), aiming to evaluate the changes in accuracy and CE loss for different subgroups.

Results

The accuracy (Acc), Cross-Entropy (CE) and misclassification costs (Cost) are reported in Table 1. Each sample is associated with a subgroup \( m \), following its probability estimation at epoch 1. The result of the best CNN in epoch 2, with the preferable CE and cost values for each subgroup \( m \) is shown in bold. We observe that (1) the CNN with the proposed AdaCSL algorithm achieves better CE values over all validation subgroups with improvements in the range of 1.57-12.28%, with an average of 5.98%. (2) the CNN with the proposed AdaCSL was further improved the misclassification costs (and accuracy) in three out of four of the subgroups (subgroups \( D^v_5, D^v_6, D^v_8 \)) with the highest local misclassification costs in Epoch 1 (marked with an underline). In the fourth subgroup \( D^v_7 \), the CNN with the AdaCSL achieves identical misclassification cost to the CNN without the AdaCSL. (3) The overall costs of the CNN without AdaCSL was 10% higher than the CNN with the AdaCSL. These results illustrate the algorithm’s novelty in resolving local high misclassification costs in the validation data set by adjusting the loss function between two consecutive learning epochs.

| Subgroup Range | Subgroup Size | Epoch 1 CNN | Epoch 2 CNN w/o AdaCSL | Epoch 2 CNN w/ AdaCSL |
|----------------|---------------|-------------|-------------------------|-----------------------|
|                |               | Val Acc     | Val Cost                | Train Acc Val Acc     | Train CE Val CE       | Train Cost Val Cost   |
| 2              | 0.1 – 0.2     | 3 100%      | 0                       | 100% 100% 0.104       | 100% 100% 0.086       | 100% 100% 0.093       |
| 3              | 0.2 – 0.3     | 6 100%      | 0                       | 98.4% 100% 0.168      | 98.4% 100% 0.188      | 98.4% 100% 0.151      |
| 4              | 0.3 – 0.4     | 32 93.8%    | 4                       | 90.3% 96.9% 0.330     | 90.3% 93.8% 0.314     | 90.3% 93.8% 0.314     |
| 5              | 0.4 – 0.5     | 78 70.5%    | 46                      | 86.4% 92.3% 0.445     | 85.9% 93.6% 0.374     | 85.9% 93.6% 0.374     |
| 6              | 0.5 – 0.6     | 120 60.0%   | 96                      | 94.1% 93.3% 0.458     | 94.1% 93.3% 0.451     | 94.1% 93.3% 0.451     |
| 7              | 0.6 – 0.7     | 100 65.0%   | 77                      | 89.5% 89.0% 0.460     | 92.5% 91.0% 0.452     | 92.5% 91.0% 0.452     |
| 8              | 0.7 – 0.8     | 55 67.3%    | 26                      | 81.6% 76.4% 0.467     | 82.9% 80.0% 0.452     | 82.9% 80.0% 0.452     |
| 9              | 0.8 – 0.9     | 6 16.7%     | 10                      | 71.4% 16.7% 0.471     | 71.4% 16.7% 0.452     | 71.4% 16.7% 0.452     |

Table 1: A comparison of CE loss, accuracy, and cost in subgroups levels between CNN with AdaCSL and without AdaCSL.

4.2 Evaluation of the AdaCSL Algorithm

We evaluate the proposed approach on four binary-class data sets, with the imbalance ratio ranging from 1 (class-balanced problem) to 8 (class-imbalanced problem), between the majority (negative) and minority (positive) samples, as follows:
4.2.1 Data sets, Experimental Setup, and Method

Melanoma Classification: Harvard Dataverse (HAM10K) (Tschandl et al., 2018) This melanoma data set consists of 1,113 melanoma positive and 8,902 melanoma negative images (1:8 ratio).

Breast Histopathology Images: (IDC) (Janowczyk and Madabhushi, 2016) Invasive Ductal Carcinoma (IDC) is the most common subtype of all breast cancers. This breast cancer data set consists of 78,786 IDC positive and 198,738 IDC negative images (1:2.5 ratio).

Real and Fake Face Detection (RFFD) (Yonsei-University, 2019) This data set comprises expert-generated high-quality 'photoshopped' face images. The data set consists of 960 positive (real) images and 960 negative (fake) images (1:1 ratio).

PlantVillage-Data Set (PV-D) (Mohanty et al., 2016) This data set comprises images of healthy and infected leaves of crops plants. The data set consists of 58,299 negative (diseased) and 27,866 positive (healthy) images (1:2.1 ratio).

We evaluate the performance of the proposed algorithm to achieve minimum costs in three setups for each data set, reflecting different ratios between misclassification costs, \( \rho \in \{ \frac{|D^-|}{|D^+|}, 3 \left( \frac{|D^-|}{2|D^+|} \right), 5 \left( \frac{|D^-|}{3|D^+|} \right) \} \), where \( |D^+| \) and \( |D^-| \) are the number of positive and negative samples in the data set, respectively. In Table 1, for example, for the data set 'HAM10K', with a ratio between classes of \( \frac{|D^-|}{|D^+|} = 8 \), we define the following three setups: \( \rho = \frac{C_{(0,1)}}{C_{(1,0)}} = 8/24/40 \).

We apply the proposed AdaCSL algorithm while obtaining a single threshold \( T_v,a \) from the entire validation data set (equivalent to assuming equal thresholds for all subgroups, \( T_{v,a}^{m} = T_{v,a}, \forall m \)).

When experimenting with the balanced data set (RFFD), not all algorithms are applied. CNN with SMOTE is redundant in all the setups. Also, CNN with TA and CNN with WCE for the experimental setup of \( \rho = 1 \), are redundant since they are equivalent to the standard CNN.

4.2.2 Results

We select the model of each compared method that yields the best misclassification costs over the validation set, and report their costs and accuracy over the test set in Table 2. The results of the
Datasets | Experimental Setup | CNN | CNN w/ TA | CNN w/ SMOTE | CNN w/ WCE | CNN w/AdaCSL
--- | --- | --- | --- | --- | --- | ---
HAM10K | $\rho = 8$ | 1001 | 72.01% | 1132 | 84.84% | 824 | 74.19% | 658 | 82.57%
 | $\rho = 24$ | 1895 | 83.25% | 2000 | 11.83% | 2198 | 85.98% | 1090 | 65.75% | 959 | 69.25%
 | $\rho = 40$ | 6012 | 88.71% | 2000 | 11.83% | 4649 | 86.33% | 1400 | 57.80% | 1345 | 60.25%
IDC | $\rho = 2.5$ | 1296 | 86.66% | 1482 | 86.12% | 1263 | 85.85% | 1249 | 84.84% | 1225 | 83.64%
 | $\rho = 7.5$ | 2712 | 85.36% | 3492 | 86.00% | 2557 | 83.93% | 1990 | 74.92% | 1925 | 78.23%
 | $\rho = 12.5$ | 4741 | 86.25% | 5250 | 86.11% | 3910 | 84.33% | 2481 | 66.79% | 2318 | 75.18%
RFFD | $\rho = 1$ | 214 | 62.85% | - | - | - | - | - | - | - | -
 | $\rho = 3$ | 322 | 59.38% | 492 | 60.07% | - | - | - | - | 210 | 63.54%
 | $\rho = 5$ | 540 | 65.97% | 538 | 62.15% | - | - | 318 | 51.73% | 274 | 57.29%
PV-D | $\rho = 2.1$ | 406 | 98.40% | 12925 | 32.34% | 428 | 98.40% | 390 | 98.22% | 170 | 99.50%
 | $\rho = 6.3$ | 1319 | 98.35% | 12925 | 32.34% | 1098 | 98.41% | 1134 | 94.26% | 279 | 98.82%
 | $\rho = 10.5$ | 1779 | 97.27% | 12925 | 32.34% | 1224 | 98.42% | 1835 | 94.12% | 585 | 98.03%

Table 2: A comparison of the performance of the proposed adaptive algorithm vs. other baseline and other cost sensitive methods.

best model are shown in bold for each data set and experimental setup. We observe that the CNN with the proposed AdaCSL algorithm achieves the best performance over all data sets and setups with improvements in the range of 2-75%, with an average of 20%. Furthermore, although the proposed algorithm is designed to yield good cost results, it achieved competitive accuracy results for experimental setups that reflect an imbalance ratio equal to the cost ratio (cost-sensitive approach for imbalance problems) with the best accuracy in two of the four data sets. Figure 1(a) shows the values of the actual threshold $T^v,a$, which yields the minimum costs in the IDC validation data set, over different epochs. It can be observed that the actual threshold converges to the desired threshold $T' = 0.5$, while differences between the imbalance ratio and cost ratio yield higher oscillation in early epochs. Finally, we observe a smooth reduction in training and validation costs for all data sets and experimental settings and Figure 1(b) show the observed decrease on the IDC data set.

5 Conclusions

We proposed a novel cost-sensitive learning method that combines both algorithmic and decision level approaches, aiming to achieve minimum misclassification costs derived from the consequences of various errors. The adaptive cost-sensitive learning (AdaCSL) algorithm was developed to apply the proposed method and adaptively adjusts the loss function such that the classifier would be directed to reduce local high misclassification costs in the validation data set. Our results on four popular data sets, with different setups, show that a CNN classifier with the AdaCSL algorithm outperformed other alternative methods, with an average cost reduction of 20%. Furthermore, the proposed algorithm achieved competitive accuracy in experimental setups that reflect an imbalance ratio equal to the cost ratio (i.e., the conventional class-imbalance problems), with better accuracy results in the class-balanced data set. Future works can apply the proposed cost-sensitive learning method and AdaCSL algorithm to design a variety of cost-sensitive learning models with different loss functions. Another possible research direction is to exploit the method and adjust the AdaCSL algorithm to multi-class classification tasks.
References

Aurelio, Y. S., de Almeida, G. M., de Castro, C. L., and Braga, A. P. Learning from imbalanced data sets with weighted cross-entropy function. *Neural Processing Letters*, 50(2):1937–1949, 2019.

Buda, M., Maki, A., and Mazurowski, M. A. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018.

Chaabane, I., Guermazi, R., and Hammami, M. Enhancing techniques for learning decision trees from imbalanced data. *Advances in Data Analysis and Classification*, pages 1–69, 2019.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee, 2009.

Domingos, P. Metacost: A general method for making classifiers cost-sensitive. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 155–164, 1999.

Dong, H., Zhu, B., and Zhang, J. A cost-sensitive active learning for imbalance data with uncertainty and diversity combination. In *Proceedings of the 2020 12th International Conference on Machine Learning and Computing*, pages 218–224, 2020.

Elkan, C. The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, volume 17, pages 973–978. Lawrence Erlbaum Associates Ltd, 2001.

Geng, Y. and Luo, X. Cost-sensitive convolution based neural networks for imbalanced time-series classification. *arXiv preprint arXiv:1801.04396*, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Hernández-Orallo, J., Flach, P., and Ferri, C. Roc curves in cost space. *Machine learning*, 93(1):71–91, 2013.

Janowczyk, A. and Madabhushi, A. Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases. *Journal of pathology informatics*, 7, 2016.

Khan, S. H., Hayat, M., Bennamoun, M., Sohel, F. A., and Togneri, R. Cost-sensitive learning of deep feature representations from imbalanced data. *IEEE transactions on neural networks and learning systems*, 29(8):3573–3587, 2017.

Kong, Q., Xu, Y., Wang, W., and Plumbley, M. D. Sound event detection of weakly labelled data with cnn-transformer and automatic threshold optimization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2450–2460, 2020.

Lan, M., Luo, J., Chai, S., Chai, R., Zhang, C., and Zhang, B. A novel industrial intrusion detection method based on threshold-optimized cnn-bilstm-attention using roc curve. In *2020 39th Chinese Control Conference (CCC)*, pages 7384–7389. IEEE, 2020.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

Ling, C. X. and Sheng, V. S. Cost-sensitive learning and the class imbalance problem. *Encyclopedia of machine learning*, 2011:231–235, 2008.

Ling, C. X., Yang, Q., Wang, J., and Zhang, S. Decision trees with minimal costs. In *Proceedings of the twenty-first international conference on Machine learning*, page 69, 2004.

Maimon, O. Z. and Rokach, L. *Data mining with decision trees: theory and applications*, volume 81. World scientific, 2014.

Masnadi-Shirazi, H. and Vasconcelos, N. Cost-sensitive boosting. *IEEE Transactions on pattern analysis and machine intelligence*, 33(2):294–309, 2010.
Mohanty, S. P., Hughes, D. P., and Salathé, M. Using deep learning for image-based plant disease detection. *Frontiers in plant science*, 7:1419, 2016.

Pandey, S. K. and Janghel, R. R. Automatic detection of arrhythmia from imbalanced ecg database using cnn model with smote. *Australasian physical & engineering sciences in medicine*, 42(4): 1129–1139, 2019.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

Raj, V., Magg, S., and Wermter, S. Towards effective classification of imbalanced data with convolutional neural networks. In *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, pages 150–162. Springer, 2016.

Ribeiro, M. T., Singh, S., and Guestrin, C. "why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

Sze-To, A. and Wong, A. K. A weight-selection strategy on training deep neural networks for imbalanced classification. In *International Conference Image Analysis and Recognition*, pages 3–10. Springer, 2017.

Ting, K. M. An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering*, 14(3):659–665, 2002.

Tschandl, P., Rosendahl, C., and Kittler, H. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data*, 5(1):1–9, 2018.

Wang, S., Liu, W., Wu, J., Cao, L., Meng, Q., and Kennedy, P. J. Training deep neural networks on imbalanced data sets. In *2016 international joint conference on neural networks (IJCNN)*, pages 4368–4374. IEEE, 2016.

Wu, Z., Guo, Y., Lin, W., Yu, S., and Ji, Y. A weighted deep representation learning model for imbalanced fault diagnosis in cyber-physical systems. *Sensors*, 18(4):1096, 2018.

Yonsei-University. Real and fake face detection, 2019. URL [https://www.kaggle.com/ciplab/real-and-fake-face-detection/](https://www.kaggle.com/ciplab/real-and-fake-face-detection/).

Zhang, C., Tan, K. C., Li, H., and Hong, G. S. A cost-sensitive deep belief network for imbalanced classification. *IEEE transactions on neural networks and learning systems*, 30(1):109–122, 2018.

Zhao, H. and Li, X. A cost sensitive decision tree algorithm based on weighted class distribution with batch deleting attribute mechanism. *Information Sciences*, 378:303–316, 2017.