1-2021

# Dynamically Weighted Balanced Loss: Class Imbalanced Learning and Confidence Calibration of Deep Neural Networks

K. Ruwani M. Fernando
*University of South Florida*, kfernando@usf.edu

Chris P. Tsokos
*University of South Florida*, ctsokos@usf.edu

# Dynamically Weighted Balanced Loss: Class Imbalanced Learning & Confidence Calibration of Deep Neural Networks

K. Ruwani M. Fernando, *Member, IEEE* and Chris P. Tsokos

***Abstract*—Imbalanced class distribution is an inherent problem in many real-world classification tasks where the minority class is the class of interest. Many conventional statistical and machine learning classification algorithms are subject to frequency bias and learning discriminating boundaries between the minority and majority classes could be challenging. To address the class distribution imbalance in deep learning, we propose a class re-balancing strategy based on a class-balanced dynamically weighted loss function where weights are assigned based on class frequency and predicted probability of ground truth class. The ability of dynamic weighting scheme to self-adapt its weights depending on the prediction scores allows the model to adjust for instances with varying levels of difficulty resulting in gradient updates driven by hard minority class samples. We further show that the proposed loss function is classification calibrated. Experiments conducted on highly imbalanced data across different applications of cyber intrusion detection (CICIDS2017 dataset) and medical imaging (ISIC2019 dataset) show robust generalization. Theoretical results supported by superior empirical performance provide justification for the validity of the proposed Dynamically Weighted Balanced (DWB) Loss Function.**

***Index Terms*—Convolutional neural networks (CNNs), cost sensitive learning, confidence calibration, data imbalance, loss functions.**

## I. Introduction

WITH ARTIFICIAL Intelligence, mobile and Internet of Things (IoT) driving data complexity and new sources of data, a new paradigm named big data has emerged. High class imbalance, often observed in large-scale datasets [1], occurs when some classes are under-represented (minority) compared with few classes that dominate (majority), introducing a distributional bias in favor of the majority classes. Such a skewed class distribution with a biased learning process could result in underestimation of minority class conditional probabilities hindering classification performance. Despite decades of research, training unbiased models from highly imbalanced datasets continues to be an open problem.

Data distribution imbalance is predominant in many real-world classification tasks, such as fault diagnosis [2], [3], network intrusion detection [4], medical diagnosis [5], [6], electricity pilferage [7] and fraudulent transactions [8], among others. Handling class-imbalance is of great importance in these situations, where the minority class is the class of interest with respect to the learning task. For instance, a malicious program should not be misclassified as benign which could lead to more adverse consequences than the reverse. Similarly, a malignant skin lesion which is rare should still be correctly identified. The same applies to several other application areas, where the accurate detection of rare events is crucial. As conventional classifiers rely on balanced class distributions, they will tend to misclassify minority observations of data with a skewed class distribution [9], [10]. Thus, a classifier which perform well and learn effectively from inherently more difficult and rare classes is highly desirable.

Obtaining reliable probability estimates is crucial to make informed decisions in real-world applications. This is even more necessary for imbalanced data due to high uncertainty around rare events. While classification of data featuring high class imbalance has received attention in prior research, reliability of class membership probabilities in the presence of class imbalance has been previously assessed only to a very limited extent [11], [12]. A closer look to the previous studies on probability calibration shows that research on classification calibration under class imbalance in the context of deep learning is so far lacking in the scientific literature.

Deep Learning (DL) has arguably become the most crucial breakthrough in machine learning and has achieved the state-of-the-art performance in various applications. Deep Neural Networks (DNNs) are comprised of sums of non-linearly transformed linear models [13] and, thus, are trained to approximate non-linear functions between the input and output. In neural networks, the feedback generated by the loss function helps in optimizing the parameters. Due to the feasibility in differentiable optimization, the most common choice for the loss function in multi-class classification is the cross entropy. Classical cross-entropy based loss function gives equal importance for each data instance, which will lead the network oversee classes with fewer number of observations. Thus, cross entropy loss is improper in classification or segmentation tasks under class imbalance. A simple heuristic method which is widely adopted to balance loss in the presence of class imbalance is to set class weights inversely proportional to the class frequency [3], [14]. However, this strategy reveals poor performance on large-scale real-world data. In contrast, we propose a dynamic strategy to assign class weights with emphasis on hard to train instances and propose a novel loss function called *Dynamically Weighted Balance (DWB) Loss* which is capable of naturally handling the class imbalance while also leading to improved calibration performance. To illustrate the generality of the proposed approach, experiments

K. Ruwani M. Fernando and Chris. P. Tsokos are with the Department of Mathematics and Statistics, University of South Florida, Tampa, FL, 33620 USA (e-mail: kfernando@usf.edu).

are conducted on challenging real-world applications in cyber intrusion detection and skin lesion diagnosis.

*Contributions:* The proposed approach is distinct in two ways with respect to the previous work: (1) Instead of a fixed weighting scheme, the assigned weights self-adapts its scale based on the prediction difficulty of the data instance. (2) We link class imbalance and reliability of confidence estimates. To the best of our knowledge, prior research has not addressed both these issues in a unified approach in the context of deep learning. The paper therefore presents the following major novelties: (1) A differentiable loss formulation based on a class rebalancing strategy, where the weights are dynamically changed during the course of training. (2) A framework that allows to learn models that are already well calibrated, thus simultaneously addressing both class imbalance and reliability of class membership probabilities in deep neural networks.

The remainder of the paper is structured as follows: In Section II, we briefly discuss related work. Section III is comprised of deep neural network preliminaries. Section IV formulates the problem and presents the proposed framework. In Section V, we experimentally evaluate our approach. Section VI concludes the paper.

## II. Related Work

### A. Class Imbalance

Despite recent advances in deep learning, the research on deep neural networks to address class imbalance remain limited [15]. We briefly describe below the traditional methods and prominent work in recent years on deep imbalanced learning.

Most of the previous efforts to handle class imbalance can be divided into two categories: data-level and algorithmic-level methods. *Data-level* methods [16]–[23] alter the class distribution in the original data by employing re-sampling strategies to balance the dataset. The simplest forms of re-sampling include random over-sampling and random under-sampling. The former handles class imbalance by duplicating the instances in the rare minority class and thus, augmenting the minority class, whereas the latter randomly drops instances from the majority class to match the cardinality of minority class. Experiments conducted in [24] suggest that data sampling strategies have little effect on classification performance, however, results in [25] demonstrate that random over-sampling leads to performance improvements. While sampling strategies are widely adopted, these methods manipulate the original class representation of the given domain and introduce drawbacks. Particularly, over-sampling can potentially lead to overfitting and may aggravate the computational burden while under-sampling may eliminate useful information that could be vital for the induction process. Moreover, a classifier developed by employing sampling methods to artificially balance data may not be applicable to a population with a much difference prevalence rate since the classifier is trained to perform well on balanced data.

*Algorithm-level* approach involve adjusting the classifier, and can further be categorized into *ensemble methods* and *cost-sensitive methods*. The most widely used methods include bagging [26] and boosting [27] ensemble-based methods. Boosting algorithms such as AdaBoost work by placing more emphasize on harder to train examples and using them to train subsequent classifiers. Experiments in [28] suggests boosting performs better than sampling methods. Alternatively, hybrid ensemble methods which combine sampling and boosting methods [29], [30] have also been proposed in past literature. A thorough review on ensemble techniques for imbalanced data with emphasis on two-class problems is presented in [31]. While ensemble-based algorithms are worthwhile, the use of multiple classifiers makes them more complex which leads to increased training times.

To reinforce the sensitivity of the classification algorithm towards the under-represented class, *cost sensitive learning* methods incorporate class-wise costs into the objective function of the classification algorithm during training process. Cost parameters can be arranged in the form of a cost matrix such that higher costs are associated with misclassification of an observation from the minority class [32]. However, design of the cost matrix which includes different misclassification costs associated with each class sample may require expert judgement. Another approach to cost-sensitive learning is rescaling the data, performed by assigning training examples of different classes with different weights (re-weighting), re-sampling the training instances or shifting the decision threshold based on their misclassification costs. These methods have been reported to perform well on binary data [32]. In [10], authors study techniques which are proven to be efficient in handling class imbalance. They conclude that while almost all methods are effective on binary classification, some methods are only effective in binary case and that cost sensitive learning can become highly complicated in multi-class setting.

Among recent contributions in *deep imbalanced learning,* Khan et al. [33] proposed a cost sensitive approach where they optimized both the model parameters and cost parameters synchronously. In the domain of computer vision, a recently proposed loss function called Focal Loss [34] for object detection attracted considerable attention in which they promote harder samples by down-weighting the loss assigned to well-classified instances. A meta-learning approach that determines per-sample loss weights of the training data based on their gradient directions is presented in [35], but requires an additional validation set and takes approximately three times the training time compared to regular training. Zhang et al. [36] proposed an evolutionary cost-sensitive deep belief network (ECS-DBN) to improve the imbalance classification performance of Deep Belief Networks (DBN). However, their approach is prohibitively expensive since the class-dependent misclassification costs are first optimized by an adaptive differential evolution algorithm (EA). A method that combines hard sample mining with a newly introduced class rectification loss (CRL) function is proposed in [37]. They adopt a batch-wise hard sample mining approach on the minority class. In [38], loss reweighting is performed by the inverse effective number of samples. Based on the assumption that the samples with too many similar gradient norms are the easy samples, authors in [39] suggested a counting based approach called Gradient Harmonizing Mechanism (GHM).

Current approaches for handling class imbalance in deep learning contains drawbacks with respect to over-fitting, loss of information, complexity and require changes to the network architectures and optimization process. Furthermore, existing methods for loss reweighting require careful tuning of hyper-parameters which can be computationally expensive.

### B. Confidence Calibration

Most of the previous studies have almost exclusively focused on either class imbalance or obtaining calibrated probability estimates, but handling both these issues concurrently remains briefly addressed in literature [11], [12]. In [11], authors show that probability estimates of the instances in minority classes are unreliable and that the methods of handling class imbalance do not automatically address calibration. Moreover, experiments in [12] demonstrates that strategies adopted to mitigate effects of class imbalance such as under-sampling adversely affect probability calibration of minority classes. In the context of neural networks, post-hoc calibration methods including matrix scaling, vector scaling and temperature scaling are widely adopted for probability calibration. Temperature scaling, proposed more recently by Guo et al. [40] gained significant attention. The method is applied to the logits of the neural network and require a validation set to tune a temperature parameter. However, the performance of these approaches in the presence of class imbalance is not adequately explored.

Differing from previous methods which require a hand-crafted cost matrix, assign fixed weights, or involve algorithmic modifications, we propose a loss function incorporating a dynamic weighting factor adjusted during the training process to address training bias of imbalanced data which also result in well-calibrated confidence estimations. It does not require any additional hyper-parameter tuning and can be promptly applied to any deep neural network architecture.

## III. Deep Neural Network Preliminaries

Through several underlying network blocks or layers, Deep Neural Networks (DNNs) extract representative features and hidden structural knowledge from data automatically. A brief description on the training paradigm of DNNs is presented below.

### A. Deep Neural Network (DNN)

In the supervised setting, in a training set with $n$ training examples $\{\mathbf{x}_i, y_i\}_{i=1}^{n}$ , each input vector $\mathbf{x}_i \in \mathbb{R}^n$ is associated with a corresponding class label (classification target) $y_i \in \{1, \ldots, c\}$. Given a feature vector $\mathbf{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d$ with $d$ individual features $x_i$ , a deep neural network with $H$ hidden layers can be represented by a non-linear function $f_\theta \colon \mathcal{X} \to \mathcal{Y}$ with model parameters $\theta = \{\theta_1, \ldots, \theta_H\}$. Here, $\theta_i = \{W_i, b_i\}$, where $W_i$ is the weight matrix and $b_i$ is the bias vector for layer $i$. Then the DNN presents a complex feature transformation through $a(\mathbf{x}) = g(W_H.g(\ldots g(W_2.g(W_1.\mathbf{x} + b_1) + b_2)\ldots) + b_H)$. Typically, the mapping function $g(\cdot)$ consists of an affine transformation (either matrix multiplication or convolution) and a non-linear transformation (activation function). The general activation formula for the $h^{\text{th}}$ layer in the $j^{\text{th}}$ node can then be represented by:

$$a_j^{[h]} = g^{[h]}\Big(\sum_k w_{jk}^{[h]} a_k^{[h-1]} + b_j^{[h]}\Big) \tag{1}$$

where $a_j^{[h]}$ is the activation of the $j^{th}$ neuron in the $h^{th}$ layer, $g^{[h]}$ is the activation function in the $h^{th}$ layer, $w_{jk}^{[h]}$ is the weight connection in the $h^{th}$ layer from neuron $j$ in $(h-1)^{th}$ layer to neuron $k$ in $h^{th}$ layer and $b_j^{[h]}$ is the bias term of the $j^{th}$ node in $h^{th}$ layer.

The feature vector in the last hidden layer is mapped to the output space $\mathcal{Y}$ to obtain the network output which is passed through a softmax function to convert into normalized (pseudo) probabilities for different possible output classes. In a softmax layer with $c$ neurons, the probability of class $j$ given the feature vector $\mathbf{x}$ is computed as:

$$P(y = j|\mathbf{x}) = \frac{\exp(a(\mathbf{x})^T W_j^s + b_j^s)}{\sum_{j=1}^{c} \exp(a(\mathbf{x})^T W_j^s + b_j^s)} \tag{2}$$

where $a(\mathbf{x})$ is the output of penultimate layer, and $W_j^s$ and $b_j^s$ are weights and bias terms in the $j^{th}$ node connecting penultimate layer to the softmax layer, $s$.

To find the optimal model parameters, the network is then updated iteratively with respect to a loss function $\mathcal{L}(f(\mathbf{x}_i; \theta), y_i)$ using an optimizer (traditionally, back-propagation algorithm):

$$\arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(\mathbf{x}_i; \theta), y_i) \tag{3}$$

where $\theta$ represents model parameters, $n$ is the sample size and $\mathcal{L}$ is the loss function.

The predicted class label $\hat{y}$ for any input instance $\tilde{\mathbf{x}}$, is the index of the maximum predicted score among all classes, $\arg\max_j [P(y = j|\tilde{\mathbf{x}})]$. The loss function for multi-class classification is usually the categorical Cross Entropy (CE) which is defined as:

$$\mathcal{L}_{CE}(\hat{y}, y) = -\sum_{j=1}^{c} y_j \log(\hat{y}_j) \tag{4}$$

where $y_j = 1$ if training instance $\mathbf{x}_i$ belongs to class $c_j$ and 0, otherwise. Particularly, the objective function CE tries to maximize the likelihood of the target class for each training instance.

### B. Convolutional Neural Network (CNN)

Deep learning models with different model architectures lend themselves to solve a large variety of problems. While 2D-CNN models have become the de facto standard for image processing applications, 1D-CNN models have shown to be effective in various applications in sequence processing such as anomaly detection [41], speech processing [42] and biomedical data classification [43].

The key attribute of neural networks is their ability to derive complex feature representations as linear combinations of the inputs which are then used to model the target as a

non-linear function of the derived features. As in traditional machine learning, deep neural network based solutions do not require application of feature engineering techniques since the feature learning process is completed automatically. Through convolutional learning and spatial pooling operations, CNNs aggregate local features to extract complex hierarchical feature representations from feature sequence.

CNNs are composed of two distinct alternating layer types: convolutional and sub-sampling layers. The first convolutional layer in a CNN extract primitive features of network traffic while the subsequent convolutional layers can deduce more sophisticated features. The activation unit in a CNN represents the results of the convolution operation of the input data with a kernel. The convolution layer is followed by a max-pooling layer for dimensionality reduction of data. Finally, the dense layer classifies the output classes combining all complex features identified by convolutional layers. More formally, feature map extraction using a one-dimensional convolution operation can be expressed as:

$$a_j^{l+1}(\tau) = \sigma\left(\sum_{f=1}^{F^l} K_{jf}^l(\tau) * a_f^l(\tau) + b_j^l\right) \tag{5}$$

where the feature map $j$ in layer $l$ is denoted by $a_j^l(\tau)$, non-linear function by $(\sigma)$, the number of feature maps in layer $l$ by $F^l$, convolution kernels by $K_{jf}^l$ and bias vector by $b_j$.

## IV. Dynamically Weighted Balanced (DWB) Loss

### A. Loss Function Formulation

*Revisiting Categorical Cross Entropy:* Let the training set with $n$ samples be denoted by $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n \subset \mathbb{R}^{d_x} \times \mathbb{R}^{d_y}$, where $\mathcal{X} \subset \mathbb{R}^{d_x}$ is the feature space and $\mathcal{Y} \subset \mathbb{R}^{d_y}$ is the label space. For each data instance $i$, $\mathbf{x}_i \subset \mathcal{X}$ is the input feature vector and $y_i \subset \mathcal{Y} = \{1, 2, \ldots, c\}$ is the ground-truth class label. Consider a hypothesis (classifier) from a parametric family $\mathcal{F} \coloneqq \{f_\theta \colon \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} | \theta \in \Theta\}$ which maps input feature space to the label space $f : \mathcal{X} \to \mathcal{Y}$ and learns by minimizing the loss $\mathcal{L}(f(\mathbf{x}; \theta), y)$. Given a loss function $\mathcal{L} \colon \mathbb{R} \times \mathcal{Y} \to \mathbb{R}^+$ and a classifier $f$, the (empirical) risk is defined as $R_L(f) = E_D[f(\mathbf{x}; \theta), y]$, where the expectation is with respect to the the empirical distribution, $D$.

Consider a DNN with the softmax output layer with loss as the categorical cross entropy. Then the parameters of DNN can be optimized with empirical risk minimization where risk is defined as:

$$R_L(f) = E_D[f(\mathbf{x}; \theta), y_\mathbf{x}] = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c} y_{ij}\ \log\left(f_j(\mathbf{x}_i; \theta)\right) \tag{6}$$

where $\theta$ is the set of parameters of the classifier, $y_{ij}$ is the $j^{\text{th}}$ element of one-hot encoded label of the instance $\mathbf{x}_i$ with $\mathbf{y}_i = e_{y_i} \in \{0, 1\}^c$ such that $\mathbf{1}^T \mathbf{y}_i = 1, \forall i$, and $f_j(\mathbf{x}; \theta) \in \mathbb{R}^c$ is the model output with $f_j$ denoting the $j^{\text{th}}$ element of $f$. Since the output layer is a softmax, $\sum_{j=1}^n f_j(\mathbf{x}_i; \theta) = 1$ and $f_j(\mathbf{x}_i; \theta) \geq 0, \forall j, i, \theta$.

*Dynamic Weighting of Loss Function:* The backpropagation of error algorithm which is typically used to train neural networks updates the weights of the model in proportion to the errors made during training. As the misclassification errors of data instances from each class are given the same importance, for severely skewed class distributions this results in adapting the classifier in favor of majority class. While class imbalance does not hinder model performance in simple classification tasks with clear class separation, it affects classes that are inherently more difficult to classify. Training samples from classes with fewer observations producing lower class probabilities are expected to be the harder instances. Moreover, correct classifications tend to have greater softmax probabilities than those misclassified and out-of-distribution instances [44]. In this context, we introduce a dynamic weighting based classifier objective function based on the prediction probability of ground truth class to assign higher weights to hard to train instances, which we term the Dynamically Weighted Balanced (DWB) Loss. Let $f_j(\mathbf{x}_i; \theta)$ be indicated by $p_{ij}$ for convenience. Thus, $p_{ij}$ is the predicted probability of the class $j$ of instance $\mathbf{x}_i$. We define Dynamically Weighted Balanced (DWB) Loss as:

$$L_{DWB} = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c} w_j^{(1-p_{ij})}\ y_{ij}\ \log\left(p_{ij}\right) - p_{ij}(1 - p_{ij}) \tag{7}$$

where $w_j$ is the class weight of class $j$, $y_{ij}$ is the $j^{\text{th}}$ element of one-hot encoded label of instance $\mathbf{x}_i$ and $p_{ij}$ is the predicted probability of the class $j$ of instance $\mathbf{x}_i$.

The proposed loss function is composed of two terms: dynamically weighted cross entropy and a regularization component equal to the entropy of brier score which can be considered as a reliability component that leads to better calibration (more on calibration is in section IV-B).

The class weights $w_j$ can be handled as a hyper-parameter that is learned from data by cross validation or set proportional to inverse class frequency. We set $w_j$ equal to the log ratio of the class frequency of the majority class and the class frequency $n_j$ (computed over the training dataset) as follows:

$$w_j = \log\left(\frac{\max(n_j | j \in c)}{n_j}\right) + 1 \tag{8}$$

As such, misclassification errors for a class $j$ with class-wise cost of $w_j$ will have $w_j$-times more penalty than misclassification errors for the majority class with weight equals to 1. For extremely imbalance classes, log smooths the weights and to avoid major class weight being less than 1, we add 1 to the log weights.

While a fixed-weighting approach based on class frequency balances the contribution from majority and minority classes, it does not discriminate between the easy and hard sample instances. Instead, we apply class-wise weights of various magnitudes from the same class depending on the prediction output and adjust the relative contribution of mispredictions. The loss function defined in equation (7) optimizes a dynamically weighted training loss which reflects labels' importance level based on class frequency while promoting hard positives which are predictions with low confidence scores.

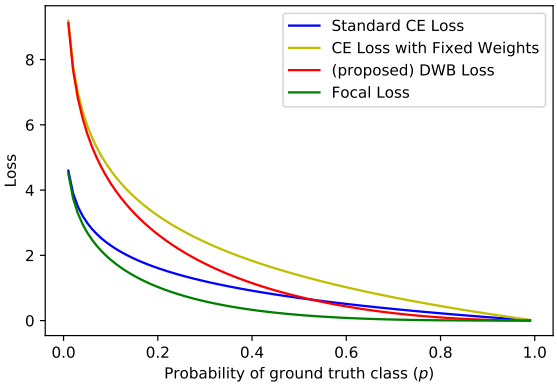


Fig. 1. Comparisons among proposed Dynamically Weighted Balanced (DWB) Loss and other commonly used losses for classification: the standard Cross Entropy (CE) loss, cross entropy with fixed weights assigned and the Focal Loss (FL) with hyper-parameter($\gamma$) set to 2 (recommended). DWB Loss put more focus on hard to train, misclassified examples through a dynamic weighting factor.

For illustration purposes, we consider a case where class weight or class imbalance ratio, $w_j = 2$. Fig. 1 provides an intuitive comparison of different losses: standard binary Cross Entropy (CE), cross entropy with fixed class weights set to imbalance ratio, Focal Loss (FL) and proposed Dynamically Weighted Balanced (DWB) Loss. It depicts how the proposed DWB loss reshapes the loss function based on the prediction probability of the target by dynamically assigning the importance weights. Note that the Focal Loss always produces a lower loss value when compared with the standard cross entropy loss. This results in FL still down-weighting correct predictions with low prediction scores ($p < 0.6$). On the contrary, the proposed DWB loss penalizes more than the cross entropy if the predictions defined from the network outputs are confident and wrong.

We note two properties of the DWB Loss: (1) When a training instance is misclassified and $p_{ij}$ is small, the loss is up-weighted. (2) As $p_{ij}$ goes close to 1, the weighting factor for well classified instance is close to 1, hence the loss is unaffected and equivalent to Cross Entropy. Differing from FL which down-weights the contribution of easy samples, proposed DWB Loss focus more on hard examples by up-weighting the misclassified examples while taking into account both sample difficulty and the class frequency. Experiments suggest that the performance of the proposed loss function is superior to the previous class balancing approaches, implying that it is a more effective alternative to the existing methods.

We visualize dynamic class weights (dash lines) in Fig. 2 for each class in imbalanced CICIDS2017 dataset assigning different predicted probabilities for ground truth. Note that $p_{ij} = 1$ corresponds to no re-weighting and $p_{ij} = 0$ corresponds to re-weighting by imbalance ratio ($w_j$) which is proportional to inverse class frequency (logarithm was not taken when computing the weighting factor in Fig. 2 for better illustration). Thus, the introduced self-adapting weighting scheme enables smooth adjustment of the class-balanced term between re-weighting and no re-weighting of objective function.

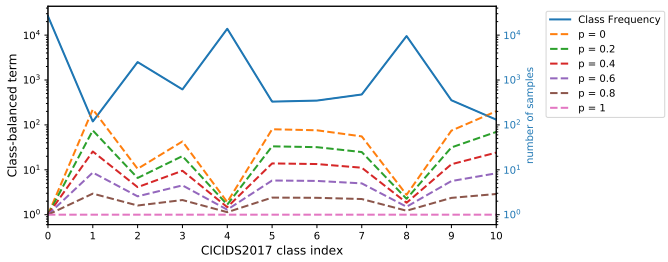


Fig. 2. Visualization of the weight term based on predicted probability of ground truth class ($p$) on long-tailed CICIDS2017 data. Y-axis is in log scale. Solid blue line represents the number of samples in each class while the dash line represents how the assigned weight changes w.r.t prediction probability of ground truth class. Note that here we have not taken logarithm when computing the weighting factor for better visualization.

### B. Improving Calibration using DWB loss

Biased training data with a skewed class distribution typically under-estimates the class probability estimates of minority class instances [11], and therefore, the predicted class probabilities are unreliable in class imbalance scenarios. The parameter estimation bias under class imbalance also applies to models which typically produce calibrated probability estimates, such as logistic regression [45]. Obtaining well-calibrated probability estimates which are reflective of the true likelihood of events [46] is highly desirable in real-world applications. The calibrated prediction probabilities are in concordance with the true occurrence of the event of interest and perfect calibration is formally defined as:

$$\mathbb{P}(Y = y|\hat{p} = p) = p \quad ;\forall p \in [0, 1] \tag{9}$$

where $Y$ is a class prediction and $\hat{p}$ is its associated confidence.

The regularizing component of the DWB loss is equal to the entropy of conditional distribution $p = p_\theta(y|\mathbf{x})$ in Brier Score. Recall that entropy of a probability assignment is a measure of inherent uncertainty [47]. Below we show that the DWB Loss minimizes a regularized upper bound on the weighted Kullback-Leibler (KL) Divergence [48] between the true distribution $\boldsymbol{q}$ and the predicted distribution $\boldsymbol{p}$.

Considering a data instance with class label $y$, ground truth probability $q_y$ and class membership probability estimate $p_y$, we proceed to obtain the following:

$$\begin{aligned}
L_{DWB} &= -w_y^{(1-p_y)}\, q_y\, \log\left(p_y\right) - p_y\left(1-p_y\right) \\
&\geq -w_y\left(1-p_y\right) q_y\, \log\left(p_y\right) - p_y\left(1-p_y\right) \\
&\qquad\qquad ;\forall y,\ w_y \geq 1 \ and\ p_y \in [0,1] \\
&= -w_y\, q_y\, \log\left(p_y\right) - w_y \mid p_y\, q_y\, \log\left(p_y\right) \mid - p_y\left(1-p_y\right) \\
&\qquad\qquad ;\forall y,\ \log\left(p_y\right) \leq 0 \\
&\geq -w_y\, q_y\, \log\left(p_y\right) - \max(q_y)\, w_y \mid p_y\, \log\left(p_y\right) \mid \\
&\qquad\qquad - p_y\left(1-p_y\right) \\
&\geq -w_y\, q_y\, \log\left(p_y\right) + w_y\, p_y\, \log\left(p_y\right) - p_y\left(1-p_y\right) \\
&\qquad\qquad ;\forall y,\ q_y \in [0,1] \\
&\geq \boldsymbol{w}\left(CE(\boldsymbol{q},\boldsymbol{p}) - H(\boldsymbol{p})\right) - \boldsymbol{p}\left(\boldsymbol{1}-\boldsymbol{p}\right)
\end{aligned} \tag{10}$$

where $CE(\boldsymbol{q},\boldsymbol{p})$ is the cross entropy between true distribution $\boldsymbol{q}$ and predicted distribution $\boldsymbol{p}$, and $H(\boldsymbol{p})$ is the entropy of $\boldsymbol{p}$.

Since, $CE(\boldsymbol{q},\boldsymbol{p}) = KL(\boldsymbol{q}||\boldsymbol{p}) + H(\boldsymbol{q})$, the above inequality can be represented as:

$$\begin{aligned} L_{DWB} &\geq \boldsymbol{w}\left(KL(\boldsymbol{q}||\boldsymbol{p}) + \underbrace{H(\boldsymbol{q})}_{\text{constant}} - H(\boldsymbol{p})\right) - \boldsymbol{p}\left(\mathbf{1}-\boldsymbol{p}\right) \\ &\geq \boldsymbol{w}\left(KL(\boldsymbol{q}||\boldsymbol{p}) - H(\boldsymbol{p})\right) - \boldsymbol{p}\left(\mathbf{1}-\boldsymbol{p}\right) \end{aligned} \tag{11}$$

where $KL(\boldsymbol{q}||\boldsymbol{p})$ represents the KL divergence between target $\boldsymbol{q}$ and predicted $\boldsymbol{p}$ distributions.

The proposed loss constructs an upper bound on the weighted KL divergence with an additional regularization equal to the sum of $\boldsymbol{w}\,H(\boldsymbol{p})$ and $\boldsymbol{p}(\mathbf{1}-\boldsymbol{p})$. While it seeks to minimize the deviation of the predicted distribution from the true label distribution through KL divergence, it aims to maximize the entropy terms, thereby penalizing over-confident predictions on the target as a form of regularization which leads to better calibration. While FL has shown to have calibration properties in [49], we did not observe significantly improved results with it in our experiments.

### C. DWB Loss Function Gradients

Let the predicted (unnormalized) output from the model be denoted by $z_i$, where $i \in \{1,...,c\}$. The softmax function $\mathbb{R}^c \rightarrow \mathbb{R}^c$, maps a vector $z \in \mathbb{R}^c$ to a vector $p \in \mathbb{R}^c$ which can be expressed as:

$$p_i(z) = \frac{e^{z_i}}{\sum_{j\in\{1,...,c\}} e^{z_j}} \quad ;\forall i \in \{1,...,c\} \tag{12}$$

where $z$ is a real vector.

Given that for a data instance with class label $y$, the only non-zero element of the one-hot encoded vector $\mathbf{y}$ is at the $y$ index, the DWB loss is simplified as:

$$L_{DWB} = -w_y^{(1-p_y)}\,\log\left(p_y\right) - p_y\left(1-p_y\right) \tag{13}$$

To check the impact of weighting factor on gradient updates, consider the first component of the DWB loss, $L1_{DWB} = -w_y^{(1-p_y)}\log(p_y)$. It is equivalent to cross entropy loss when $w = 1$, for which the loss function gradients are as follows:

$$L_{CE} = -\log\left(p_y\right) = -\log\left(\frac{e^{z_y}}{\sum_j e^{z_j}}\right) \tag{14}$$

$$\begin{aligned} \nabla_{z_i} L_{CE} &= \nabla_{z_i}\left(-z_y + \log\sum_j e^{z_j}\right) \\ &= \frac{1}{\sum_j e^{z_j}}\nabla_{z_i}\sum_j e^{z_j} - \nabla_{z_i} z_y \\ &= p_i - \nabla_{z_i} z_y \\ &= p_i - \mathbb{1}(y=i) \end{aligned} \tag{15}$$

where

$$\mathbb{1}(y=i) = \begin{cases} 1 & ;y=i \\ 0 & ;otherwise \end{cases}$$

When $w_y \neq 1$, the gradients of the dynamic weighting factor $w^{(1-p_y)}$ reduces to:

$$\begin{aligned} \nabla_{z_i} w_y^{(1-p_y)} &= \nabla_{z_i} w_y^{\left(1-\frac{e^{z_y}}{\sum_j e^{z_j}}\right)} \\ &= -w_y^{\left(1-\frac{e^{z_y}}{\sum_j e^{z_j}}\right)}\log\left(w_y\right)\nabla_{z_i}\left(1-\frac{e^{z_y}}{\sum_j e^{z_j}}\right) \\ &= w_y^{(1-p_y)}\log\left(w_y\right)\left[p_y\,\mathbb{1}(y=i) - p_y\,p_i\right] \end{aligned} \tag{16}$$

Using the product rule, we obtain the gradients of $L1_{DWB}$ as follows:

$$\nabla_{z_i} L1_{DWB} = w_y^{(1-p_y)}[1+p_y\,\log\left(p_y\right)\,\log\left(w_y\right)][p_i - \mathbb{1}(y=i)] \tag{17}$$

Thus, when compared with cross entropy loss, the DWB loss weights each data instance by an additional weighting factor. Consequently, the predictions that are less congruent with the provided ground-truth labels are weighed more in the gradient update, which in turn provides more emphasis on neural network training of difficult samples.

## V. Experiments

### A. Experimental set-up and Evaluation

*Experiments:* We evaluate the proposed approach on two challenging real-world tasks: Cyber-Intrusion Detection and Skin Lesion Diagnosis, and a detailed description of each is provided in subsequent sections. The following loss functions are compared in terms of classification and calibration performance: 1) *Cross entropy* is set as the baseline, 2) *Weighted Cross Entropy* weights each data instance by the inverse frequency, 3) *Focal Loss* down-weighs the easy samples, 4) (Proposed) *DWB Loss* dynamically weights loss contribution of each data instance focusing on hard to train instances.

*Classification Evaluation:* In an extreme class imbalanced setting, a classifier that simply predicts any instance as belonging to the majority class could achieve a deceptively high accuracy. We evaluate the model classification performance subject to four different metrics: Precision, Recall/Sensitivity (Detection rate), F-measure and AUROC Score. Let us define a particular class $j$ as a positive instance and all other classes as negatives. The performance metrics for a particular class label $(j)$ are defined as follows:

$$\begin{aligned} Precision_j(Pr) &= TP_j/(TP_j + FP_j) \\ Recall_j(Re) &= TP_j/(TP_j + FN_j) \\ F1-score_j &= (2\times Pr \times Re)/(Pr + Re) \end{aligned} \tag{18}$$

where $TP$ are True positives, $TN$ are True Negatives, $FP$ are False Positives and $FN$ are False Negatives.

Precision reflects the proportion of a specific label classified correctly with respect to instances which were predicted to belong in that class. Recall is defined as the proportion of instances that are predicted to belong to a class and truly belong in the class. F1-Score is the weighted harmonic mean of precision and recall. The average of the recall of each class is equivalent to balanced multi-class accuracy. In addition to aforementioned classification metrics, we utilized Area

Under the Receiver Operating Characteristics (AUROC) as an evaluation criteria.

*Calibration Metrics:* We evaluate the calibration performance based on Expected Calibration Error (ECE) [50], Maximum Calibration Error (MCE) and Brier Score (BS) [51]. While ECE is the most common calibration metric, it has several drawbacks [52]. We use BS as the primary metric for calibration evaluation which measures the average squared loss between the estimated class membership probabilities and true class value. Lower values indicate better calibration. BS is formally defined as follows:

$$BS = 1/n \sum_{i=1}^{n} \sum_{j=1}^{c} (y_{ij} - p_{ij})^2 \tag{19}$$

where $n$ is the overall number of instances, with $y_{ij}$ and $p_{ij}$ denoting the $j^{\text{th}}$ element of one-hot encoded class label and predicted probability of the instance $\mathbf{x}_i$, respectively.

### B. Experiment 1: Cyber Intrusion Detection

An Intrusion Detection System (IDS) dynamically monitors network traffic to efficiently detect cyber-attacks from normal legitimate traffic [53]. As network intrusions represent only a small subset of all network traffic, the size of the benign traffic outweighs that of the malicious traffic. The fact that the overwhelming majority of network traffic will be in the 'benign' class and rare positive cases (malicious network traffic) will be in the 'attack' class, create an extreme class imbalance problem. Intrusion detection can therefore be interpreted as a multi-class classification problem under high class imbalance.

*1) Dataset Description:* We rely on an intrusion detection dataset (CICIDS2017) published by the Canadian Institute for Cyber-Security (CIC) at the University of New Brunswick (UNB) [54]. Captured network flow records in the dataset resembles the real-world network traffic and include both normal and malicious attack traces. This flow-based data is captured within a five-day timeframe in 2017 and contains 3.1 million flow records. Each network flow record is characterized by 86 features which can be categorized into *time-based features* (e.g. flow duration and inter-arrival packet time), *size of payload data* (e.g. total application bytes and maximum size of the packets) and *packet count* (e.g. source to destination packet count). Certain attack classes in the dataset are highly underrepresented categorizing intrusion detection for CICIDIS2017 as an extremely imbalanced multi-class classification problem.

*2) Implementation (Intrusion Detection System Model Overview):* While deep learning models can extract features automatically, conventional machine learning classifiers involve a feature selection phase and hence, implemented under three stages: (a) *Pre-processing phase*: includes data cleaning, stratified Train-Validation-Test split procedure and data transformations; (b) *Feature Selection phase*: Implementation of correlation analysis followed by feature selection through Recursive Feature Elimination with Cross Validation (RFE-CV); (c) *Classification phase*: involves model fitting and performance evaluation. Classification performance of conventional classifiers and deep neural networks with different loss functions is then compared.

TABLE I
THE DISTRIBUTION OF NETWORK FLOWS IN EACH ATTACK CATEGORY

| Class Category | Count | Percentage(%) |
|---|---|---|
| Benign | 44002 | 48.2780 |
| DoS Hulk | 23107 | 25.3525 |
| PortScan | 15893 | 17.4374 |
| DDoS | 4183 | 4.5895 |
| DoS GoldenEye | 1029 | 1.1290 |
| FTP-Patator | 794 | 0.8712 |
| SSH-Patator | 590 | 0.6473 |
| DoS slowloris | 580 | 0.6364 |
| DoS Slowhttptest | 550 | 0.6034 |
| Web Attack | 218 | 0.2392 |
| Bot | 197 | 0.2161 |

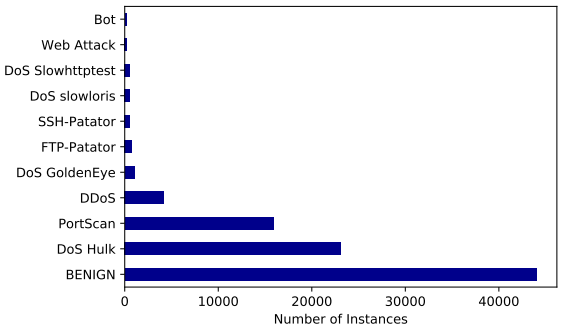


Fig. 3. Network activity flow distribution with network flow-count varying sharply across different attack categories.

*Data Pre-Processing Phase:* The dataset is comprised of separate attack files for each attack class. We therefore, first combined all attack records into Denial of Service (DoS) attack file which encompasses the largest number of Benign records. However, the prevalence rate of each attack in individual data files remain approximately the same after merging them. Two attack types (Heartbleed and Infiltration) were omitted since they constitute only a very small fraction of flow records. Individual web attack classes were merged together into a single web attack category. Nominal Features that are related to a specific network and another ten features that contained all zero entries were removed from the data frame. After the pre-processing stage the data-frame dimension reduces to 911421 records with 66 network flow feature variables. For training purposes, we only considered 10% of the data (stratified sample). The network activity flow distribution across different attack categories after pre-processing stage is depicted in Table I and Fig. 3. Three subsets were obtained in a stratified manner for training (60%), validation (20%) and testing (20%) purposes. Stratification enables to randomly split the dataset while retaining the correct class distribution in each subset, which is the recommended way of splitting data under class imbalance [55]. In order to avoid biased results, feature vector is transformed by scaling each feature to a [0,1] range. Categorical variable 'class label' was transformed through one-hot encoding.

*Feature Selection Phase for conventional classifiers:* For conventional Machine Learning (ML) classifiers, we conducted a feature selection procedure to identify representative

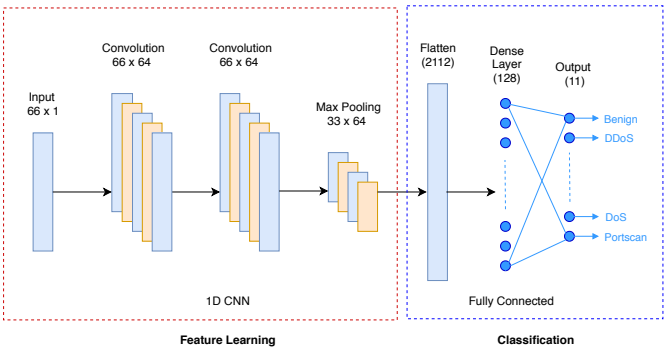


Fig. 4. 1D-CNN Model Architecture. It includes two convolution layers and a pooling layer followed by a standard fully connected neural network.

TABLE II
CICIDS2017 DATASET: AVERAGE METRIC VALUES (PERCENTAGES)

| classification Algorithm | Precision | Recall | F1-score | AUROC Score |
|---|---|---|---|---|
| **Conventional ML Classifiers** | | | | |
| Multinomial Logistic Regression | 37.75 | 22.26 | 24.67 | 58.89 |
| Decision Tree | 94.43 | 95.75 | 95.05 | 97.85 |
| Random Forest | 95.57 | 95.66 | 95.60 | 97.80 |
| XGboost | 95.90 | 95.31 | 95.51 | 97.61 |
| Gradient Boosting | 91.55 | 91.36 | 91.02 | 95.64 |
| **DL: 1D CNN Model** | | | | |
| CE Loss | 97.15 | 96.00 | 96.50 | 97.97 |
| Weighted CE | 97.38 | 97.44 | 97.40 | 98.69 |
| Focal Loss | 96.96 | 98.05 | 97.49 | 98.89 |
| DWB Loss | **97.52** | 98.00 | **97.74** | **98.99** |

and distinguishable features for intrusion detection. We first conducted a correlation analysis to identify possible correlations. Considering 0.90 as the correlation coefficient threshold, 32 features with a correlation magnitude greater than 0.90 were removed. Then, an optimal subset of features was obtained through Recursive Feature Elimination with Cross-Validation (RFE-CV) which is used to train classical ML classifiers. The selected subset containing 11 features includes the total number of data packets in the forward direction, the total quantity of bytes in the forward direction, the maximum and mean values of the packet's length in bytes in the forward direction, the maximum value in bytes of the packet's length in the backward direction, the mean and standard deviation of the inter-arrival time of the flow in both directions, the number of packets per second in the backward direction, the minimum length of the packets registered in the flow in both directions, the total number of bytes sent in the initial window in the forward and backward directions.

*1D Convolutional Neural Net (1D-CNN) Model:* The one-dimensional convolution neural network based intrusion detector had the best performance compared with other DL models. The implementation of cost sensitive classification with CNN does not require a feature selection phase since convolution layers are capable of extracting better representations from data automatically. We consider 1D-CNN with convolutions in the spatial domain. The applicability of the 1D-CNN model in CICIDS2017 network flow data can be justified as follows: We notice that there is high correlation among features and data contain features that belong to similar groupings. Thus, there is a local pattern in the features and the relative spatial positioning of the data is relevant with local relationships in data providing more predictive information for the classification task. Hence, the idea of local spatial correlation in CNN translates well to the problem at hand. We expect interesting features to depend on short consecutive sub-sequences of the input. We treat the input features as spatial dimension and the kernel is convolved over input features. We expect 1D-CNN model to capture specific patterns from successive input features and thus derive a more robust representation of features which contain important information for identification of malicious network flows.

The intrusion detector 1D-CNN model architecture is depicted in Fig. 4. It involves an input layer (shape 66 x 1) , two convolutional layers with one-dimensional filter kernels of size 3, max pooling layer with sub-sampling factor 2, a flattening layer, one dense layer and a final output layer with the number of nodes equal to number of classes. The activation function of the hidden dense layers is Rectified Linear Unit (ReLU) and Softmax is employed in the output layer for the multi-class classification. Each network is trained for 200 epochs with Adam optimization [56] method.

*3) Experimental Results:* Using the reduced feature subset obtained in the feature selection stage, we tested several widely used traditional machine learning classifiers including Multinomial Logistic Regression, Random Forests, Decision Tree, Gradient boosting and XGBoost. Their performance in terms of average precision, recall and F1-Score is presented in Table II with multinomial logistic regression having the worst performance. This result is not surprising and in consistent with the previous research which have proven the performance degradation of conventional logistic regression under class imbalance [45]. Except for multinomial logistic regression which is highly affected by the imbalanced class distribution, other conventional classification algorithms seem to be performing well. However, their performance is comparatively low in comparison to DL Models. While we experimented with several different DNN and 1D-CNN model architectures, we only included the results of the best performing 1D-CNN model in the paper. The average classification results in Table II, as well as the class-wise classification performance of 1D-CNN Model trained with different loss functions in Table III suggest that the proposed DWB loss clearly outperforms the other commonly used objective functions in cost sensitive learning. Specifically, F1 score and recall or the 'Detection Rate' of attacks is highest with the proposed method for the most extremely imbalanced classes, such as Bot attacks which occupy only 0.2% of data.

For CICIDS2017 data we provide only the results of our primary calibration metric, Brier Score since values for other calibration metrics are extremely small that the difference is insignificant (Table IV). The Brier Score is at its lowest when trained with the proposed DWB loss function implying better calibration.

TABLE III
CICIDS2017 DATASET: CLASS-WISE CLASSIFICATION PERFORMANCE

| | | Benign | Bot | DDoS | DoS GoldenEye | DoS Hulk | DoS Slowhttptest | DoS Slowloris | FTP Patator | PortScan | SSH Patator | Web Attack |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Precision | CE Loss | 99.49 | 84.38 | 100 | 98.54 | 99.65 | 95.5 | 99.13 | 99.37 | 100 | 99.14 | 93.48 |
| | Weighted CE | 99.87 | 85.71 | 99.76 | 97.15 | 99.14 | 96.43 | 99.13 | 99.37 | 100 | 96.61 | 93.33 |
| | Focal Loss | 99.87 | 85.71 | 99.76 | 97.14 | 99.14 | 96.43 | 99.13 | 99.37 | 100 | 96.61 | 93.34 |
| | DWB Loss | 99.82 | 88.1 | 99.88 | 99.5 | 99.5 | 95.5 | 99.13 | 100 | 100 | 100 | 91.3 |
| Recall | CE Loss | 99.67 | 69.23 | 99.88 | 98.54 | 99.5 | 96.36 | 98.28 | 99.37 | 99.97 | 97.46 | 97.72 |
| | Weighted CE | 99.31 | 92.31 | 100 | 99.03 | 100 | 98.18 | 98.28 | 99.37 | 99.96 | 96.61 | 95.45 |
| | Focal Loss | 99.31 | 92.31 | 100 | 99.03 | 100 | 98.18 | 98.28 | 99.37 | 99.97 | 96.61 | 95.45 |
| | DWB Loss | 99.61 | 94.87 | 99.88 | 97.57 | 100 | 96.36 | 98.28 | 99.37 | 99.96 | 96.61 | 95.45 |
| F1-Score | CE Loss | 99.58 | 76.06 | 99.94 | 98.54 | 99.58 | 95.93 | 98.7 | 99.37 | 99.98 | 98.29 | 95.56 |
| | Weighted CE | 99.67 | 91.14 | 99.7 | 97.8 | 99.8 | 96.83 | 98.7 | 98.4 | 99.98 | 98.29 | 91.11 |
| | Focal Loss | 99.59 | 88.88 | 99.88 | 98.08 | 99.57 | 97.3 | 98.7 | 99.37 | 99.98 | 96.61 | 94.38 |
| | DWB Loss | **99.72** | **91.36** | 99.88 | **98.54** | 99.75 | 95.93 | **98.7** | **99.68** | **99.98** | **98.29** | 93.33 |

TABLE IV
CICIDS2017 DATASET: CALIBRATION PERFORMANCE

| | CE Loss | Weighted CE | Focal Loss | DWB Loss |
|---|---|---|---|---|
| Brier Score | 0.0067 | 0.0065 | 0.0116 | **0.0056** |

### C. Experiment 2: Skin Lesion Diagnosis

Skin lesions are among the most common cancers worldwide with over 5,000,000 newly identified cases in the United States every year. Melanoma is the deadliest skin malignancy, but if diagnosed early has a survival rate which exceeds 95%. To facilitate early and accurate detection of skin cancers, a fast and automated diagnosis system is crucial. Dermoscopy is a skin imaging modality which is pivotal in detection of skin malignancies and supports towards implementation of automated algorithmic systems. Lesion detection is one of the most challenging tasks in medical imaging due to high similarity between lesions and intra-class variations with respect to texture, color, size, shape and location. Class imbalanced nature of the diagnosis task makes it even more challenging.

*1) Dataset Description:* We utilize ISIC2019 challenge skin lesion data [57]–[59], published by International Skin Imaging Collaboration (ISIC) in ISIC Archive which is the largest publicly available repository of dermosocopic images. The goal is to classify skin lesions based on 25,3331 dermoscopy images available for training which are unequally distributed among 8 different lesion categories. The skin lesion diagnosis distribution is presented in Table V and Fig. 5.

*2) Implementation Details:* To ensure the class distribution remains the same, we split ISIC2019 data to train-test-validation subsets in a stratified manner such that train data contains 19,173 data entries, with validation and test sets each having 1070 unique entries. Both skin lesion dermoscopic image data and meta data were employed for lesion detection model implementation following a dual input strategy (Fig. 6). Meta data contains patient age, gender and anatomy site. Meta and dermoscopy data were pre-processed prior to training and images were augmented during training with random flipping, color, shift and rotation transformations to ensure robustness to deformations, thereby better generalization.

TABLE V
THE DISTRIBUTION OF SKIN LESION DIAGNOSTIC CATEGORY

| Diagnosis Category | Count | Percentage(%) |
|---|---|---|
| Melanocytic nevi (NV) | 12875 | 50.83 |
| Melanoma (MEL) | 4522 | 17.85 |
| Basal cell carcinoma (BCC) | 3323 | 13.12 |
| Benign keratosis (BKL) | 2624 | 10.36 |
| Actinic keratosis (AK) | 867 | 3.42 |
| Squamous cell carcinoma (SCC) | 628 | 2.48 |
| Vascular lesion (VASC) | 253 | 1.00 |
| Dermatofibroma (DF) | 239 | 0.94 |

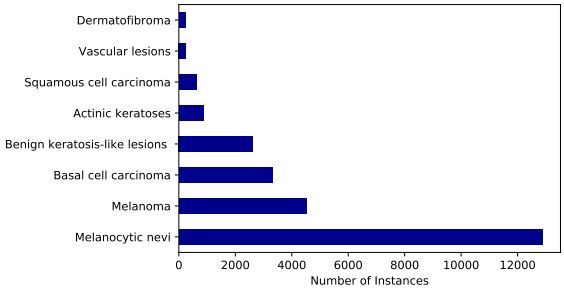


Fig. 5. Skin Lesion Diagnosis distribution with lesion count varying sharply across different diagnosis categories.

*Model Architecture and Training:* We relied on established methods for computer vision and used the state-of-the-art CNN models for image classification. We chose EfficientNet (EN-B3) [60] as the model architecture since it performed significantly better than the other experimented models. To be consistent with the chosen ImageNet pre-trained model, the input images were resized to 300 x 300. Each network was trained for 30 epochs with Stochastic Gradient Descent (SGD) [61] optimization algorithm.

*3) Experimental Results:* We evaluate the impact of different loss functions for training to diagnose skin lesions and the result of this analysis is presented in Table VI and Table VII. Average classification metric values of diagnosis categories and class-wise classification performance suggest that the DWB loss considerably outperforms the other loss functions in terms of classification.

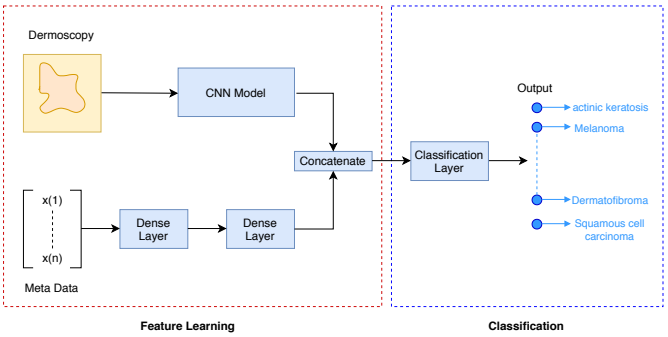


Fig. 6. Schematic diagram of the dual-input neural network model architecture composed of a 2D-CNN (EfficientNet-B3) and fully connected model.

TABLE VI
ISIC2019 DATASET: AVERAGE METRIC VALUES (PERCENTAGES)

| Loss Function | Precision | Recall | F1-Score | AUROC |
|---|---|---|---|---|
| CE Loss | 66 | 64 | 65 | 80 |
| Weighted CE | 67 | 66 | 66 | 81 |
| Focal Loss | 64 | 60 | 61 | 78 |
| DWB Loss | **69** | **66** | **67** | **82** |

TABLE VII
ISIC2019 DATASET: CLASS-WISE CLASSIFICATION (PERCENTAGES)

| | | Skin Lesion Diagnosis Category | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | AK | BCC | BKL | DF | MEL | NV | SCC | VASC |
| Precision | CE Loss | 50 | 77 | 70 | 50 | 49 | 92 | 64 | 78 |
| | Weighted CE | 41 | 79 | 71 | 75 | 49 | 93 | 50 | 78 |
| | Focal Loss | 36 | 72 | 71 | 62 | 51 | 91 | 50 | 75 |
| | DWB Loss | 42 | 78 | 74 | 88 | 56 | 93 | 50 | 75 |
| Recall | CE Loss | 62 | 81 | 63 | 50 | 54 | 93 | 41 | 70 |
| | Weighted CE | 56 | 76 | 67 | 75 | 53 | 93 | 36 | 70 |
| | Focal Loss | 50 | 67 | 67 | 62 | 49 | 93 | 27 | 60 |
| | DWB Loss | 46 | 79 | 70 | 88 | 60 | 93 | 35 | 67 |
| F1-Score | CE Loss | 56 | 79 | 67 | 50 | 51 | 93 | 50 | 74 |
| | Weighted CE | 47 | 77 | 69 | 75 | 51 | 93 | 42 | 74 |
| | Focal Loss | 42 | 70 | 69 | 62 | 50 | 92 | 35 | 67 |
| | DWB Loss | 46 | **79** | **70** | **88** | **60** | **93** | 35 | 67 |

Calibration Performance evaluation results for ISIC2019 data is presented in Table VIII. With DWB objective function, we observe calibration results which are much improved over the other losses trained with the same network.

TABLE VIII
ISIC2019 DATASET: CALIBRATION METRICS

| Loss Function | ECE | MCE | Brier Score |
|---|---|---|---|
| CE Loss | 0.0553 | 0.2212 | 0.2596 |
| Weighted CE | 0.0330 | 0.1321 | 0.2622 |
| Focal Loss | 0.0612 | 0.2454 | 0.2458 |
| DWB Loss | **0.0295** | **0.0938** | **0.2389** |

The experimental results are in consistent across detection tasks in different domains, implying that when trained with the proposed loss function, the model surpasses the performance of conventional classifiers in terms of both classification and calibration.

## VI. CONCLUSION

To address class imbalance encountered in many practical, real-world classification tasks, we present a self-adapting weighting approach and introduce a novel loss function, named Dynamically Weighted Balanced (DWB) Loss. Weighting scheme is based on class frequency of training data and prediction difficulty of individual data instances. The prediction difficulty is determined by the prediction score produced by the neural network. We further demonstrate that the regularization component in the proposed loss function leads to improved calibration performance. Experiments in different domains: cyber intrusion detection (tabular data) and skin lesion diagnosis (image classification) show consistent results implying robust generalization. A considerable performance improvement was observed in rare minority classes with the proposed DWB loss function over different kinds of other widely adopted loss functions when tested for the same model architecture. Presented method can be adapted for any classification or segmentation task owning broad applicability and its superior performance suggests the potential of cost-sensitive deep learning based models for real-life deployment.

## REFERENCES

[1] J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya, "A survey on addressing high-class imbalance in big data," *Journal of Big Data*, vol. 5, no. 1, p. 42, 2018.
[2] Z.-B. Zhu and Z.-H. Song, "Fault diagnosis based on imbalance modified kernel fisher discriminant analysis," *Chemical Engineering Research and Design*, vol. 88, no. 8, pp. 936–951, 2010.
[3] Z. Wu, Y. Guo, W. Lin, S. Yu, and Y. Ji, "A weighted deep representation learning model for imbalanced fault diagnosis in cyber-physical systems," *Sensors*, vol. 18, no. 4, p. 1096, 2018.
[4] D. A. Cieslak, N. V. Chawla, and A. Striegel, "Combating imbalance in network intrusion datasets." in *GrC*, 2006, pp. 732–737.
[5] M. E. Celebi, H. A. Kingravi, B. Uddin, H. Iyatomi, Y. A. Aslandogan, W. V. Stoecker, and R. H. Moss, "A methodological approach to the classification of dermoscopy images," *Computerized Medical imaging and graphics*, vol. 31, no. 6, pp. 362–373, 2007.
[6] M. A. Mazurowski, P. A. Habas, J. M. Zurada, J. Y. Lo, J. A. Baker, and G. D. Tourassi, "Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance," *Neural networks*, vol. 21, no. 2-3, pp. 427–436, 2008.
[7] M. Di Martino, F. Decia, J. Molinelli, and A. Fernández, "Improving electric fraud detection using class imbalance strategies." in *ICPRAM (2)*, 2012, pp. 135–141.
[8] M. Zareapoor, P. Shamsolmoali *et al.*, "Application of credit card fraud detection: Based on bagging ensemble classifier," *Procedia computer science*, vol. 48, no. 2015, pp. 679–685, 2015.
[9] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent data analysis*, vol. 6, no. 5, pp. 429–449, 2002.
[10] Z.-H. Zhou and X.-Y. Liu, "Training cost-sensitive neural networks with methods addressing the class imbalance problem," *IEEE Transactions on knowledge and data engineering*, vol. 18, no. 1, pp. 63–77, 2005.
[11] B. C. Wallace and I. J. Dahabreh, "Improving class probability estimates for imbalanced data," *Knowledge and information systems*, vol. 41, no. 1, pp. 33–52, 2014.
[12] A. Dal Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi, "Calibrating probability with undersampling for unbalanced classification," in *2015 IEEE Symposium Series on Computational Intelligence*. IEEE, 2015, pp. 159–166.
[13] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.
[14] Y. S. Aurelio, G. M. de Almeida, C. L. de Castro, and A. P. Braga, "Learning from imbalanced data sets with weighted cross-entropy function," *Neural Processing Letters*, vol. 50, no. 2, pp. 1937–1949, 2019.
[15] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, p. 27, 2019.

[16] M. Kubat, S. Matwin *et al.*, "Addressing the curse of imbalanced training sets: one-sided selection," in *Icml*, vol. 97. Nashville, USA, 1997, pp. 179–186.
[17] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.
[18] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD explorations newsletter*, vol. 6, no. 1, pp. 20–29, 2004.
[19] T. Jo and N. Japkowicz, "Class imbalances versus small disjuncts," *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 40–49, 2004.
[20] H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. IEEE, 2008, pp. 1322–1328.
[21] N. V. Chawla, N. Japkowicz, and A. Kotcz, "Special issue on learning from imbalanced data sets," *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 1–6, 2004.
[22] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-smote: a new over-sampling method in imbalanced data sets learning," in *International conference on intelligent computing*. Springer, 2005, pp. 878–887.
[23] T. Maciejewski and J. Stefanowski, "Local neighbourhood extension of smote for mining imbalanced data," in *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*. IEEE, 2011, pp. 104–111.
[24] C. L. Castro and A. P. Braga, "Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data," *IEEE transactions on neural networks and learning systems*, vol. 24, no. 6, pp. 888–899, 2013.
[25] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, 2018.
[26] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.
[27] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.
[28] C. Seiffert, T. M. Khoshgoftaar, and J. Van Hulse, "Improving software-quality predictions with data sampling and boosting," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 39, no. 6, pp. 1283–1294, 2009.
[29] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "Smoteboost: Improving prediction of the minority class in boosting," in *European conference on principles of data mining and knowledge discovery*. Springer, 2003, pp. 107–119.
[30] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "Rusboost: A hybrid approach to alleviating class imbalance," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 40, no. 1, pp. 185–197, 2009.
[31] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 463–484, 2011.
[32] X.-Y. Liu and Z.-H. Zhou, "The influence of class imbalance on cost-sensitive learning: An empirical study," in *Sixth International Conference on Data Mining (ICDM'06)*. IEEE, 2006, pp. 970–974.
[33] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, "Cost-sensitive learning of deep feature representations from imbalanced data," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 8, pp. 3573–3587, 2017.
[34] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.
[35] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," *arXiv preprint arXiv:1803.09050*, 2018.
[36] C. Zhang, K. C. Tan, H. Li, and G. S. Hong, "A cost-sensitive deep belief network for imbalanced classification," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 1, pp. 109–122, 2018.
[37] Q. Dong, S. Gong, and X. Zhu, "Imbalanced deep learning by minority class incremental rectification," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 6, pp. 1367–1381, 2018.
[38] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9268–9277.
[39] B. Li, Y. Liu, and X. Wang, "Gradient harmonized single-stage detector," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 8577–8584.
[40] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," *arXiv preprint arXiv:1706.04599*, 2017.
[41] U. R. Acharya, H. Fujita, O. S. Lih, Y. Hagiwara, J. H. Tan, and M. Adam, "Automated detection of arrhythmias using different intervals of tachycardia ecg segments with convolutional neural network," *Information sciences*, vol. 405, pp. 81–90, 2017.
[42] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," in *2012 IEEE international conference on Acoustics, speech and signal processing (ICASSP)*. IEEE, 2012, pp. 4277–4280.
[43] D. Li, J. Zhang, Q. Zhang, and X. Wei, "Classification of ecg signals based on 1d convolution neural network," in *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*. IEEE, 2017, pp. 1–6.
[44] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," *arXiv preprint arXiv:1610.02136*, 2016.
[45] G. King and L. Zeng, "Logistic regression in rare events data," *Political analysis*, vol. 9, no. 2, pp. 137–163, 2001.
[46] M. Kuhn, K. Johnson *et al.*, *Applied predictive modeling*. Springer, 2013, vol. 26.
[47] J. Bröcker, "Reliability, sufficiency, and the decomposition of proper scores," *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, vol. 135, no. 643, pp. 1512–1519, 2009.
[48] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.
[49] J. Mukhoti, V. Kulharia, A. Sanyal, S. Golodetz, P. H. Torr, and P. K. Dokania, "Calibrating deep neural networks using focal loss," *arXiv preprint arXiv:2002.09437*, 2020.
[50] M. P. Naeini, G. F. Cooper, and M. Hauskrecht, "Obtaining well calibrated probabilities using bayesian binning," in *Proceedings of the... AAAI Conference on Artificial Intelligence. AAAI Conference on Artificial Intelligence*, vol. 2015. NIH Public Access, 2015, p. 2901.
[51] G. W. Brier, "Verification of forecasts expressed in terms of probability," *Monthly weather review*, vol. 78, no. 1, pp. 1–3, 1950.
[52] J. Nixon, M. W. Dusenberry, L. Zhang, G. Jerfel, and D. Tran, "Measuring calibration in deep learning." in *CVPR Workshops*, 2019, pp. 38–41.
[53] S. Rodda and U. S. R. Erothi, "Class imbalance problem in the network intrusion detection systems," in *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*. IEEE, 2016, pp. 2685–2688.
[54] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." in *ICISSP*, 2018, pp. 108–116.
[55] S. Raschka, "Model evaluation, model selection, and algorithm selection in machine learning," *arXiv preprint arXiv:1811.12808*, 2018.
[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[57] P. Tschandl, C. Rosendahl, and H. Kittler, "The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific data*, vol. 5, p. 180161, 2018.
[58] N. C. Codella, D. Gutman, M. E. Celebi, B. Helba, M. A. Marchetti, S. W. Dusza, A. Kalloo, K. Liopyris, N. Mishra, H. Kittler *et al.*, "Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic)," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. IEEE, 2018, pp. 168–172.
[59] M. Combalia, N. C. Codella, V. Rotemberg, B. Helba, V. Vilaplana, O. Reiter, C. Carrera, A. Barreiro, A. C. Halpern, S. Puig *et al.*, "Bcn20000: Dermoscopic lesions in the wild," *arXiv preprint arXiv:1908.02288*, 2019.
[60] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *arXiv preprint arXiv:1905.11946*, 2019.
[61] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.