1-to-N Large Margin Classifier

Jaime Rocca Layza*, Helio Pedrini*, Ricardo da Silva Torres†
*Institute of Computing, University of Campinas, Campinas, SP, Brazil
†Department of ICT and Natural Sciences, Norwegian University of Science and Technology (NTNU), Ålesund, Norway

***Abstract*—Cross entropy with softmax is the standard loss function for classification in neural networks. However, this function can suffer from limitations on discriminative power, lack of generalization, and propensity to overfitting. In order to address these limitations, several approaches propose to enforce a margin on the top of the neural network specifically at the softmax function. In this work, we present a novel formulation that aims to produce generalization and noise label robustness not only by imposing a margin at the top of the neural network, but also by using the entire structure of the mini-batch data. Based on the distance used for SVM to obtain maximal margin, we propose a broader distance definition called 1-to-N distance and an approximated probability function as the basis for our proposed loss function. We perform empirical experimentation on MNIST, CIFAR-10, and ImageNet32 datasets to demonstrate that our loss function has better generalization and noise label robustness properties than the traditional cross entropy method, showing improvements in the following tasks: generalization robustness, robustness in noise label data, and robustness against adversarial examples attacks.**

I. Introduction

The standard loss function used in neural networks for classification is arguably the cross entropy with softmax. This approach is continually used along the time especially for image classification [1], [2]. However, recent works have pointed out many limitations when the standard cross entropy is used. Examples of limitation include the lack of representation power, propensity to overfitting, and misclassification caused by adversarial examples even when they are slightly perturbed [3]–[10].

In recent works, several alternatives [3], [4], [7]–[9] have been proposed for the cross entropy loss function based on the large margin principle. That is, the classifier seeks for the model parameters that produce the largest distance to the decision boundary. This principle has theoretical and empirical foundations and has shown desirable benefits, such as better generalization and robustness to input perturbations [11].

In general, the approaches that use the notion of margin in neural networks can be classified into two categories. In the first category, the approaches impose a margin in the output of the neural network [3], [7]–[9]. All these approaches propose to put the margin specifically at the softmax function. Thus, given a sample (x_i, y_i), the margin value decreases the probability mapping produced by the softmax function at the y_i output, making the classification more rigorous and confident when achieves or passes the margin value for such y_i output. Since these types of approaches use the last hidden layer representation and define the margin only in the output space, they are efficient in training time and viable for various practical application, such as face recognition and face verification [3], [7]–[9].

In the second category, the approaches impose margin not only on the output of the network, but also in the hidden and input layers [4]. Due to the intractability of an exact margin computation in a nonlinear boundary given by the entire neural network, the loss function is approximated. This type of approaches commonly demands large computational resources and training time due to its intrinsic complexity that increases for each hidden layer that is included [4].

In this work, we propose a novel formulation that imposes a margin value on the output of the neural network, however, unlike the previous methods from this category, we do not focus on providing a margin to the traditional softmax function. Instead, we impose margin over a new probability mapping function approximated from a broad distance (that we called 1-to-N), which is based on the distance used for SVM to obtain maximal margin.

Our contributions are summarized as:

1) A novel general loss function for classification based on the large margin principle with better generalization and noise label robustness properties than the traditional cross entropy.
2) 1-to-N distance (Equation 5) and its approximated probability mapping function that transforms the network output of a sample (x^i, y^i) to its probability using the network outputs from all its negatives samples present in the mini-batch (that is, $\{(x^j, y^j)|y^j \neq y^i\}$) (Equation 6). This approach is different from the well-know softmax function that uses the network outputs for every class that only the sample x^i produces (Equation 1). To the best of our knowledge, this type of formulation has not appeared before.
3) We provide an analytical comparison of our approach with the more related counterparts using the interpretation of the equation that describes the hyperplane orientation updated in each learning step. We also show that the hyperplane orientation equations of all related approaches are instances or variations of the SVMs orientation equation when it is solved by dual optimization [11]–[13].

The ultimate goal of a classification model is a good generalization ability that has direct implication in several real applications. In the data collection process, for example, is often susceptible to mistakes. Thus, noise label robustness approaches have become very useful. Generalization is also important in techniques, such as metric learning and few-

shot learning, where commonly they deal with limited training examples [14]. Recently, works have shown a disappointment characteristic of the neural network models when adversarial examples caused misclassification even when they are slightly perturbed. There are specific techniques for addressing adversarial examples attacks, but this is also related to the poor model generalization [10], [15].

Therefore, due to the generalization importance in a broad range of sceneries and applications, we follow the experimentation protocol of Elsayed et al. [4] to evaluate the effectiveness of our approach in the following tasks: generalization robustness using smallest training data, robustness in noise label data, and robustness against adversarial examples attacks. We show that our proposed loss function outperforms the models training with the traditional cross entropy method in all the tasks.

II. Related Work

Given the data $\{(x^i, y^i)\}_{i=1}^M$ with $x^i \in \mathbb{R}^{1\times h}$, the cross entropy loss function L_{CE} is defined as:

$$S(f_{x^i}^{Wyi}) = \frac{e^{f_{x^i}^{Wyi}}}{\sum_{k=1}^{K} e^{f_{x^i}^{Wyk}}} \tag{1}$$

$$L_{\text{CE}} = -\frac{1}{M}\sum_{i=1}^{M} \log S(f_{x^i}^{Wyi}) \tag{2}$$

where $S(f_{x^i}^{Wyi})$ is the softmax function, K is the number of classes and $f_{x^i}^{Wyi} = W_{yi}^T x^i + b_{yi}$ is the network output from the input x^i for the class y^i; with $W_{yi} \in \mathbb{R}^{h\times 1}$ and and $b_{yi} \in \mathbb{R}$.

Since the softmax function is the central piece in the L_{CE} various works [3], [7]–[9] propose to impose a margin value (δ) in order to make the classification more rigorous by decreasing the output value, that is, $f_{x^i}^{Wyi} - \delta$ (for example as in the work of Liang et al. [7]). Since the output value for the correct class y^i becomes smaller than the output for the other classes, the learning process must continue until achieving the largest output. In the same line, other works [3], [8], [9] transform the network output to the angle space to introduce an angular margin $f_{x^i}^{Wyi} = \|W_{yi}\|\|x^i\|\cos(\theta)$, where θ is the angle between x^i and W_{yi}. In this case, the classification is more rigorous when the angle θ increases. Thus, many alternatives have been proposed to increase θ, such as the product $\theta\delta$, the sum $\theta + \delta$, or some combination of them [3], [8], [9].

Another approach that can be easily implemented in the neural network output using the hinge loss function is the Soft-SVM [16]. This approach is the relaxation of the Hard-SVM constraints to allow the margin violation of some examples, which is useful for the not linearly separable cases. The margin here is defined as the minimum Euclidean distance from the samples to the hyperplane and is the same that we use for our formulation. The Soft-SVM is formulated as follows [12], [13]:

$$\hat{W_{yi}}, \hat{b_{yi}} = \min_{W_{yi}, b_{yi}} \frac{\lambda}{2}\|W_{yi}\|^2 + \frac{1}{M}\sum_{j=1}^{M}[1 - y^i(f_{x^j}^{Wyi})]_+ \tag{3}$$

where $[z]_+ = \max(0, z)$ and $\lambda > 0$ that controls the tradeoff between the norm minimization and the margin violation.

Elsayed et al. [4] introduced a loss function especially designed to explore the margin feature not only in the output space, but also in the input and any chosen set of hidden layers of a network. The margin here is defined directly in a nonlinear boundary produced by the neural network and based on any distance metric, such as l_1, l_2, and l_∞ norms. Thus, for each input sample, this formulation enforces a margin in each hidden representation computing gradient with respect to each hidden layer and for each class.

Our approach differs from the method of Elsayed et al. [4] in the sense that we only impose margin on the output of the network, which avoids computing gradients throughout the network layers that could increase the training time.

III. Proposed Approach

Given a binary classification problem, we need to find the parameters of the hyperplane $H_w : w^T x^i + b$ that separates the data $\{x^i, y^i\}_{i=1}^m$, where $x^i \in \mathbb{R}^d$, $w \in \mathbb{R}^d$ and $y^i \in \{-1, +1\}$.

The SVM approach computes the optimal hyperplane that separates the data with maximal margin by minimizing the squared norm $\|w\|^2$ or equivalent maximizing the following distance [11]:

$$\rho(w, b) = \min_{\{x^i | y^i = 1\}} \gamma^i - \max_{\{x^j | y^j = -1\}} \gamma^j \tag{4}$$

Under the constraints $y^i(w^T x^i + b) \geq 1 \;\; i = 1, ..., m$, where $\gamma^i = \frac{w^T x^i + b}{\|w\|}$ and $\gamma^j = -\frac{(w^T x^j + b)}{\|w\|}$ are the distance from the samples $(x^i, y^i = 1)$ and $(x^j, y^j = -1)$ to the hyperplane H_w.

We propose a more relaxed and broad distance, which we called 1-to-N distance, defined as:

$$\gamma_i^{\textit{1-to-N}} = \frac{w^T x^i + b}{\|w\|} - \sum_{\{x^j | y^j = -1\}} \frac{w^T x^j + b}{\|w\|} \tag{5}$$

$\gamma_i^{\textit{1-to-N}}$ computes the summation of one positive γ^i and all negative γ^j distances. It means that, whereas $\rho(w, b)$ only considers closer positive and negative samples to the hyperplane, $\gamma_i^{\textit{1-to-N}}$ considers all negative samples for one positive.

Now, we approximate a probability mapping function to the $\gamma_i^{l\text{-}to\text{-}N}$ as:

$$\begin{aligned}\gamma_i^{l\text{-}to\text{-}N} &\equiv \exp(w^T x^i + b - \textstyle\sum_{\{x^j|y^j=-1\}} w^T x^j + b),\\ &\|w\| = 1\\ &= \frac{e^{w^T x^i + b}}{\prod_{\{x^j|y^j=-1\}} e^{w^T x^j + b}}\\ &\lessapprox \frac{e^{w^T x^i + b}}{\sum_{\{x^j|y^j=-1\}} e^{w^T x^j + b}}\\ &\geq \frac{e^{w^T x^i + b}}{e^{w^T x^i + b} + \sum_{\{x^j|y^j=-1\}} e^{w^T x^j + b}}\end{aligned} \tag{6}$$

We approximate a probability mapping function "like softmax"[1] by transforming the distance to the exponential scale, then by approximating a product by the summation, and by taking a lower bound (Equation 6). Thus, we obtain a softmax's form function and guarantee a valid probability mapping (with a range in $[0, 1]$).

Then, we can maximize $\gamma_i^{l\text{-}to\text{-}N}$ distance or, equivalently, maximize its approximated probability function as:

$$\underset{i=1,\ldots,m}{\operatorname{argmax}} \frac{e^{\delta-[\delta-w^T x^i+b]_+}}{e^{\delta-[\delta-w^T x^i+b]_+} + \sum\limits_{\{x^j|y^j=-1\}} e^{-\delta+[\delta+w^T x^j+b]_+}} \tag{7}$$

Our approach maximizes the probability function over a bounded region given by the margin δ. The probability is maximum when the number of samples across the margin is minimum, which follows the same idea of Soft-SVM [11]–[13].

We proposed a broad distance and a bounded margin with the goal of gain generalization not only by imposing a margin value, but also by using the whole structure of the data.

In general, for a K class classification problem, we learn the parameters of a neural network model using the data $\{x^i, y^i\}_{i=1}^M$, where $y^i = 1, ..., K$ and $x^i \in \mathbb{R}^h$ is the output of last hidden layer. We define a "like softmax" function over the data and using a margin value δ as:

$$P(f_{x^i}^{Wyi}) = \frac{e^{\delta-[\delta-f_{x^i}^{Wyi}]_+}}{e^{\delta-[\delta-f_{x^i}^{Wyi}]_+} + \sum\limits_{\{x^j|y^j\neq y^i\}} e^{-\delta+[\delta+f_{x^j}^{Wyi}]_+}} \tag{8}$$

where $f_{x^i}^{Wyi} = W_{yi}^T x^i + b_{yi}$ is the network output for the class y^i, but here we prefer to consider as the distance of the sample x^i to the hyperplane defined by $W_{yi} \in \mathbb{R}^{h\times 1}$ and $b_{yi} \in \mathbb{R}$ with $\|W_{yi}\| = 1$. Similarly, $f_{x^j}^{Wyi} = W_{yi}^T x^j + b_{yi}$ is the distance of the sample x^j to the hyperplane defined by (W_{yi}, b_{yi}). The set $\{x^j|y^j \neq y^i\}$ is the set of samples present in the data whose class is different from y^i. Figure 2b illustrates this principle.

[1]We called "like softmax" function because it is different from the well-known softmax function derived from an exponential family distribution.

Note the difference of $P(f_{x^i}^{Wyi})$ with respect to the softmax function $S(f_{x^i}^{Wyi})$ defined in Equation 1. In addition to the bounded margin, also in its denominator, $P(f_{x^i}^{Wyi})$ computes the distance between all the samples $\{x^j|y^j \neq y^i\}$ and the hyperplane defined by (Wyi, b_{yi}), whereas $S(f_{x^i}^{Wyi})$ in its denominator computes the distance between the sample x^i and all the hyperplanes (K hyperplanes). Figure 2b illustrates this principle.

Then, we maximize the probability $P(f_{x^i}^{Wyi})$ over the data via the maximum likelihood, $\operatorname{argmax}_{W_{yi}} \prod_{i=1}^M P(f_{x^i}^{Wyi})$, which is equivalent to minimize the following loss function:

$$L_{1\text{-}to\text{-}N} = -\frac{1}{M}\sum_{i=1}^{M} \log P(f_{x^i}^{Wyi}) \tag{9}$$

We experiment with two types of bounded margin, the $[-\delta, \delta]$ used in Equation 8 and the margin $[0, \delta]$. Figure 1 illustrates the principle of how the bounded values encourage the convergence.

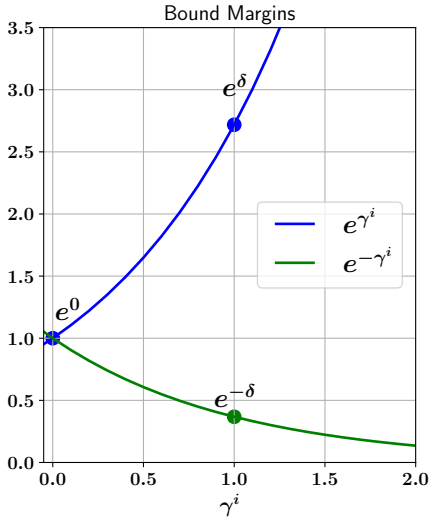


Fig. 1. For $[-\delta, \delta]$ the probability value of $P(f_{x^i}^{Wyi})$ is maximum when $\gamma^i = \delta$ and $\gamma^j = -\delta$, whereas for $[0, \delta]$ $P(f_{x^i}^{Wyi})$ is maximum when $\gamma^i = \delta$ and $\gamma^j = 0$ (along the blue line).

A. Method Comparison

Suppose the mini-batch of data $\{(x^i, y^i), (x^j, y^j)|y^i \neq y^j\}$ of some usual size (e.g., 256 samples), where y^i is fixed for some class (called "positive samples") and y^j (called "negative samples") that can take random values from the available classes $\{k\}_{k=1}^K - y^i$.

Given a hyperplane $H_{W_{yi}} : W_{yi}^T x^i + b_{yi}$ in the last layer and assuming that we use SGD for optimization, the orientation of the hyperplane will change for each iteration of the training process in order to separate the data as: $W_{yi} = W_{yi} + \alpha \frac{\partial L}{\partial W_{yi}}$, with $\frac{\partial L}{\partial W_{yi}} < 0$. It is common to compute the $\frac{\partial L}{\partial W_{yi}}$ for one sample, but it is more useful if we consider all the samples

into the mini-batch. Then, it could show a more intuitive shape of $\frac{\partial L}{\partial W_{yi}}$ to see how the SGD fits the hyperplane in order to separate the data.

We use this idea to compare our $L_{1\text{-}to\text{-}N}$ function against the most related approaches: the hinge loss (L_{Hinge}) and cross entropy (L_{CE}). Thus, we use the loss function L_{CE} for cross entropy defined in Equation 2 and the hinge loss function defined as:

$$L_{\text{Hinge}} = \sum_{\{x^i, x^j\}} [\delta - f_{x^i}^{Wyi}]_+^2 + [\delta + f_{x^j}^{Wyi}]_+^2 + \frac{\lambda}{2} \sum_{k=1}^{K} \|W_{yk}\|^2 \tag{10}$$

We compute the partial derivatives for $L_{1\text{-}to\text{-}N}$, L_{Hinge}, and L_{CE} with respect to W_{yi} using all the samples present in the mini-batch[2]:

$$\frac{\partial L_{1\text{-}to\text{-}N}}{\partial W_{yi}} = -\sum_{\{x^i\}} \sum_{\{x^j\}} P(f_{x^j}^{Wyi})(x^i - x^j) \tag{11}$$

$$\frac{\partial L_{\text{Hinge}}}{\partial W_{yi}} = -[\sum_{\{x^i\}} (\delta - f_{x^i}^{Wyi}) x^i - \sum_{\{x^j\}} (\delta + f_{x^j}^{Wyi}) x^j] \tag{12}$$

$$\frac{\partial L_{\text{CE}}}{\partial W_{yi}} = -[\sum_{\{x^i\}} \sum_{\{k \neq y^i\}} S(f_{x^i}^{Wk}) x^i - \sum_{\{x^j | y^j \neq y^i\}} S(f_{x^j}^{Wyi}) x^j] \tag{13}$$

We see that the three approaches share some variation of vector difference between the positive and negative samples, whereas for $L_{1\text{-}to\text{-}N}$ and L_{CE} the difference is weighted by their probability values, for L_{Hinge} its difference is weighted by how much the samples invade the margin. We also observe that all approaches are similar to $w = \sum_{\{y^i=+1\}} \alpha_i x^i - \sum_{\{y^j=-1\}} \alpha_j x^j$ for binary classification found in SVM when the dual optimization problem is solved [11]–[13].

The main difference between $L_{1\text{-}to\text{-}N}$ and L_{CE} is in the $P(f_{x^i}^{Wyi})$ (Equation 8) and $S(f_{x^i}^{Wyi})$ (Equation 1) computations. If we consider the softmax function as a mapping from a set of values to its probabilities equivalence, we see that $S(f_{x^i}^{Wyi})$ is computed using the set of distances from the x^i sample to every hyperplane (with normals $\{Wyk\}_{k=1}^{K}$), without taking into account rest of samples present on the batch of data (Figure 2a). Thus, the softmax function uses the distance between one sample and all hyperplane as the core of its computation.

On the other hand, $P(f_{x^i}^{Wyi})$ computes the set of distances from the x^i and $\{x^j | y^j \neq y^i\}$ samples to one hyperplane (defined by (W_{yi}, b_{yi})), as illustrated in Figure 2b. Then, $P(f_{x^i}^{Wyi})$ function uses all samples and one hyperplane as the core of its computation. Therefore, our approach gains a "broad context" that is useful for generalization and noise robustness.

Another difference between $L_{1\text{-}to\text{-}N}$ and L_{CE} is the complexity to compute $S(f_{x^i}^{Wyi})$ when the number of classes increases. L_{1-to-N} does not care about this problem due to the denominator computation in $P(f_{x^i}^{Wyi})$, which only depends on the number of samples (in the worst case, ≈ 512 mini-batch size is used in practice).

With respect to the $L_{1\text{-}to\text{-}N}$ and L_{Hinge}, we observe that the main difference is that in L_{Hinge} the distance values used to weight the samples are independent for each x^i and x^j samples (Figure 3a), whereas for $L_{1\text{-}to\text{-}N}$ the $P(f_{x^i}^{Wyi})$ value is computing relating x^i and all x^j samples.

We also show in Figure 3b the involved distance that could be used for an join approach between cross entropy and 1-to-N approaches. In Section IV, we show the results of experiments with this approach.

IV. Experiments

In this section, we present the empirical evaluation of our approach in generalization robustness, robustness in noise label data, and robustness against adversarial examples attacks.

We perform the evaluation on the following datasets: (i) MNIST [17], which contains handwritten digits (from 0 to 9) images of 28×28 shapes divided into 60,000 training images and 10,000 test images, (ii) CIFAR-10 [18] is a 10-class dataset consisting of 60,000 images of 32×32×3 shapes divided into 50,000 training images and 10,000 test images, and (iii) ImageNet32 [19] is a downsampled (images of 32×32×3 shapes) variant of the original ImageNet 2012 challenge dataset, created to facilitate fast experimentation for network architectures, training algorithms, and hyperparameters search. This dataset consists of the same number of classes than the original ImageNet (1000 classes) and samples (1,281,167 training images and 50,000 validation images).

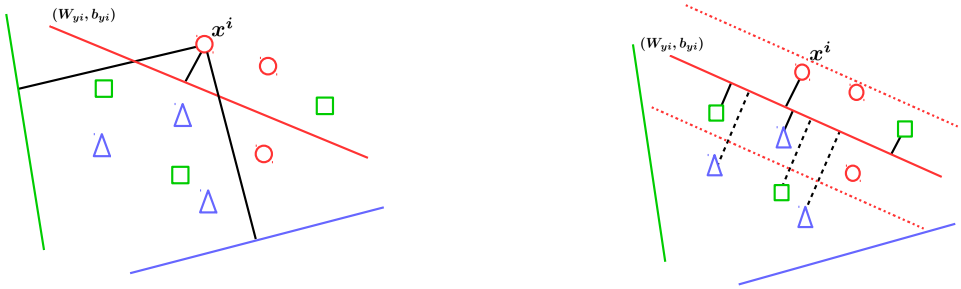
We follow the experimentation protocol of Elsayed et al. [4]. For generalization robustness, we train models with random subsets of training data of different sizes. For robustness in noise label data, we create noisy training data by introducing a different percentage of incorrect labels. For robustness against adversarial examples, we experiment with attacks in the black-box and white-box scenarios.

A. Implementation Details

We use TensorFlow 1.15.2 to implement our approach. For MNIST dataset, we train the same network used by Elsayed et al. [4], which is a 4 hidden-layer model with 2 convolutional layers. For CIFAR-10 and ImageNet32 datasets, we train a Wide Residual Network [20], specifically a WRN-28-2, where 28 is the depth value that represents the total number of convolutional layers and 2 is the widening value that multiplies the number of filters.

The $L_{1\text{-}to\text{-}N}$ is easy to implement, however, perhaps the novelty here is that, given a mini-batch of data, we need a look-up table that indicates the negative samples for each positive

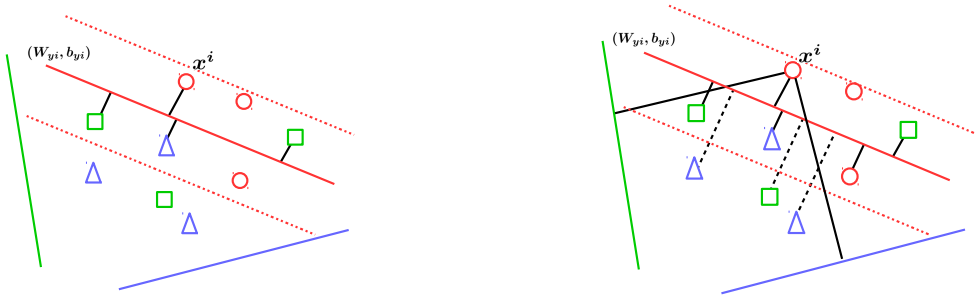
[2]For simplicity, in Equation 12, we not include the derivation of the term $\frac{\lambda}{2} \sum_{k=1}^{K} \|W_{yk}\|^2$.



(a) For Cross Entropy method.

(b) For 1-to-N approaches.

Fig. 2. Intuition about the distance (black lines) from one sample to hyperplanes (colored lines) that are used for cross entropy and 1-to-N approaches. The dashed black lines represent the bounded distances for samples out the margin.



(a) For Hinge Loss method.

(b) For 1-to-N + Cross Entropy approaches.

Fig. 3. Intuition about the distance (black lines) from one sample to hyperplanes (colored lines) that are used for hinge loss and 1-to-N + cross entropy approaches. The dashed black lines represent the bounded distances for samples out the margin.

sample. We efficiently created a look-up table as the difference of two matrices created with the labels.

For the hinge loss implementation (Equation 10), we use $\lambda = 0$ because we do not use any type of regularization method in all experiments.

For the join approach of 1-to-N and cross entropy (1-to-N+CE), we use the following probability function:

$$\frac{e^{\delta - [\delta - f_{x^i}^{W_{yi}}]_+}}{\sum\limits_{\{x^j | y^j \neq y^i\}} e^{-\delta + [\delta + f_{x^j}^{W_{yi}}]_+} + \sum\limits_{k=1}^{K} e^{f_{x^i}^{W_{yk}}}} \tag{14}$$

The probability function used for $L_{1\text{-}to\text{-}N}$ with $[0, \delta]$ bounded margin varies only in the second term of the denominator in Equation 8, replacing the term $\sum_{\{x^j | y^j \neq y^i\}} e^{-\delta + [\delta + f_{x^j}^{W_{yi}}]_+}$ by $\sum_{\{x^j | y^j \neq y^i\}} e^{[\delta + f_{x^j}^{W_{yi}}]_+}$.

The size of the margin is one more hyperparameter that depends on the dataset. Thus, we need to be careful about large margin values. For instance, if we consider infinite margin value the probability $P(f_{x^j}^{W_{yi}})$ could be maximum in the middle of the positive samples producing a bad hyperplane convergence.

B. Hyperparameter Details

For the MNIST and CIFAR-10 datasets, we train over the 90% of the training data and use the remaining 10% as validation data. For the prediction step, we use the entire test data.

In all our experiments, we use a batch size of 256, SGD with RMSProp, batch normalization, without dropout. In addition, we use data augmentation for CIFAR-10 and ImageNet32 consisting of horizontal flips, take random crops from image padded by 4 pixels on each side and mean/std normalization. We obtained accuracy results very close to the state of the art on each dataset, using the following setup for training: for the MNIST dataset, we use a learning rate of 0.01 and run for 18 epochs of training for the cross entropy and 1-to-N models, whereas we run for 30 epochs with the hinge loss model. For the CIFAR-10 dataset, we use a learning rate of 0.1 with decay of 0.5 for every 6,000 steps and we run for 60 epochs of training for the cross entropy and 1-to-N models, whereas we run for 100 epochs with the hinge loss model. For the ImageNet32, we use a learning rate of 0.01 with decay of 0.2 for every 10 epochs, and momentum of 0.9. We run for 25 epochs of training for the cross entropy and 1-to-N models, whereas we run for 30 epochs with the hinge loss model. These results are very close to the state of the art on ImageNet32 using WRN-28-2 reported by Chrabaszcz et al. [19].

The accuracy value on a particular benchmark depends on several details [21], among them, how complex the network architecture is. The WRN-28-2 used on the CIFAR-10 and ImageNet32 datasets is not a very complex model since our

main purpose is to introduce and show the advantages of our formulation against cross entropy rather than to improve the state of the art for each benchmark.

C. Generalization Robustness

Given a dataset, we randomly choose samples from the training set to create small subsets by keeping a fraction of the data. For example, on MNIST, a subset of 68 samples that represent the 0.125% is the smallest subset (Figure 4). We train (without using data augmentation) all the models using the same subset and evaluate its performance on the entire test set. With the goal to avoid bias about easy subsets that would favor the performance of some particular method, we compute the mean of the performance on 5 random subsets for each subset (for instance, 5 subsets of 68 samples).

The performance result of the 1-to-N models with different margin values for MNIST and CIFAR-10 datasets are shown in Figure 4. The accuracy curves show that the 1-to-N approaches were better than the cross entropy method especially for the smallest subsets where the generalization is more important. We did not observe conclusive results about the role of the margin value. Thus, for the MNIST dataset, the methods with large values obtained roughly higher accuracy, however, this effect was not observed on the CIFAR-10 dataset.

D. Noise Label Robustness

Given a dataset, we create noisy data by randomly choose a percentage of samples from the training set and randomly exchange its labels by incorrect values (for instance, the label '1' by the label '3'). The percentage of incorrect labels varies from 0% to 80% of the train set as used by Elsayed et al. [4]. The new noisy set is used for all methods as training data, the evaluation accuracy is computed on the entire test set when each method achieves around 80% accuracy on the noisy training set.

In a similar way to the generalization robustness experiments, we do not use data augmentation and, to avoid bias about easy noise data, we compute the mean of the performance on 5 random noisy dataset (for example, 5 noisy training sets created by change 20% of the training labels).

The performance result of the 1-to-N models with different margin values on MNIST and CIFAR-10 datasets are shown in Figure 5. The accuracy curves show that the 1-to-N approaches were better than the cross entropy method. In a similar way to the generalization results, we did not see conclusive results about the role of the margin value for the both datasets, because, on the MNIST dataset, the approaches with small margin were better than those with large margin, however, for CIFAR-10 dataset, this behavior was opposite.

E. Robustness Against Adversarial Examples Attacks

In the image data context, the adversarial examples look like part of the training or testing sets, but contain imperceptible perturbations designed to produce fool a machine learning model [10]. There are several methods to create adversarial examples, among the most commonly used are Fast Gradient Sign Method (FGSM) [5] and its iterative extension, the Basic Iterative Method [6], which can produce more rigorous perturbations by applying FGSM for multiples times. In short, we call it as IFGSM as used by Elsayed et al. [4].

Given a dataset, we use the training data to train all models. Then, we choose some model to be "the attacker," which will generate adversarial examples from the entire testing data with different levels of perturbation severity (ϵ) using the IFGSM method. Then, "the defender" model will evaluate its accuracy over the generated adversarial examples. In the black-box attack, the "the defenders" and "the attacker" models are different, whereas these models are the same as in the white-box.

We perform the black-box attack using the cross entropy approach and the defense using the 1-to-N and hinge loss models with different margin values, whereas that, for the white-box, we perform an attack and defense for each model. The performance results for MNIST and CIFAR-10 datasets are shown in Figures 6 and 7, respectively, whereas that, for ImageNet32, the performance results are shown in Figure 8.[3] The accuracy curves show that, for all attacks and all datasets, the hinge loss and 1-to-N models were superior to the cross entropy approach.

With respect to the black-box attack on the MNIST and CIFAR-10 datasets, the hinge loss and 1-to-N obtained similar accuracy, being hinge loss slightly better. The curves also show that the approaches with small margin were better than those with large margin in most of the cases. This behavior did not happen on the ImageNet32 dataset, where the large margin approach obtained better accuracy.

In the white-box attack over all datasets, the 1-to-N approaches were very superior to the hinge loss and the role of the margin was clearer than in black-box, especially on the CIFAR-10, where the approaches with large margin were better than those with smaller margin value.

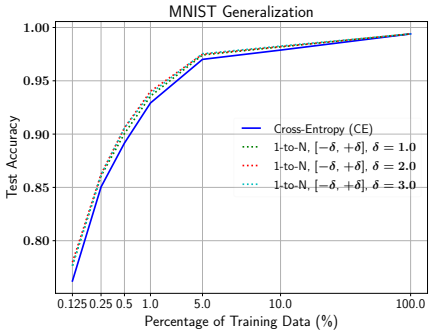
Finally, with respect to the bound margins, we see that the methods that use $[0, +\sigma]$ achieve better performance on all the datasets than those that use $[-\sigma, +\sigma]$. In addition, the join approach of 1-to-N and cross entropy (1-to-N+CE) led to a boost to the cross entropy method evidenced in the accuracy improvement obtained.

V. Conclusions and Future Work

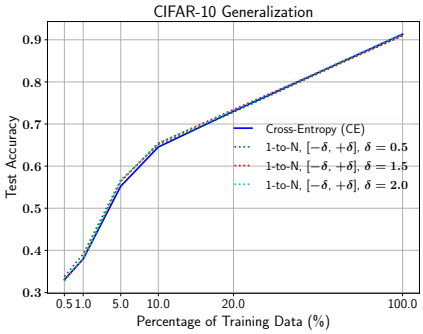
We introduced a broader idea of distance (1-to-N distance) that allows us to derive a novel probability mapping function ("like softmax"), which is the core of our loss formulation. Our probability mapping explores the samples in the mini-batch for its computation, instead of the hyperplanes for each individual sample as the traditional softmax function. This is a fundamental difference between them.

We performed empirical experiments from several variations of 1-to-N models in different useful tasks and benchmarks, showing that our formulation has better generalization and

[3]For clarity, we only show the results of three methods for the black-box attack and the methods with larger margin value for the white-box attack.

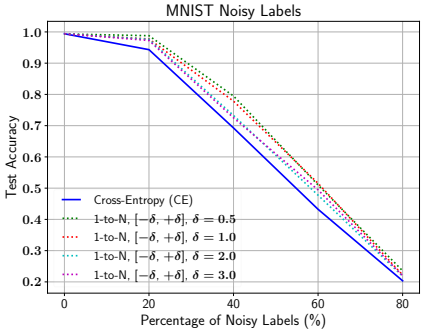


(a) Result for the MNIST dataset

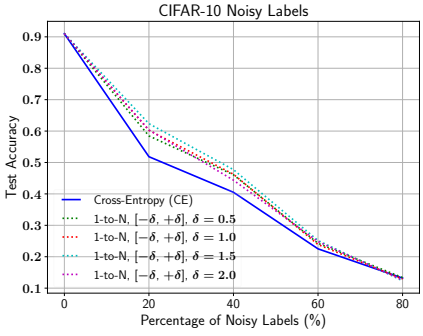


(b) Result for the CIFAR-10 dataset

Fig. 4. Performance accuracy versus subsets of training data.

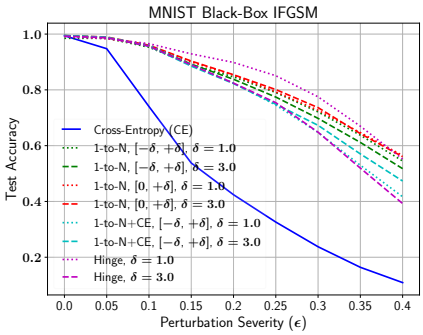


(a) Result for the MNIST dataset

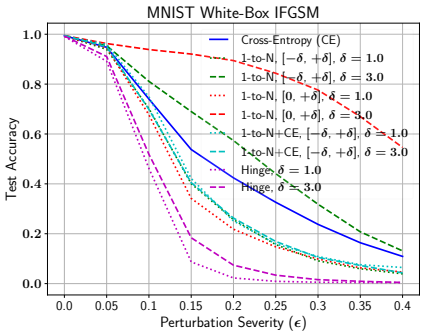


(b) Result for the CIFAR-10 dataset

Fig. 5. Performance accuracy versus different percentages of noise labels.



(a) IFGSM black-box attack



(b) IFGSM white-box attack

Fig. 6. Performance accuracy versus different values of perturbation for MNIST dataset.

noise label robustness properties than the cross entropy. In addition, the margin value makes it flexible to generalize over adversarial example attacks. Therefore, for large margin values, our approach is robust to white-box attacks, whereas it is robust for black-box attacks for small values.

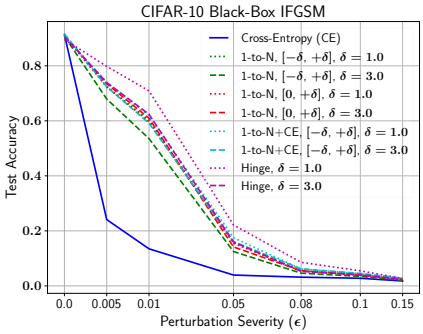
As directions for future work, we intend to improve and extend the 1-to-N formulation for many applications in different classification scenarios.

VI. Acknowledgments

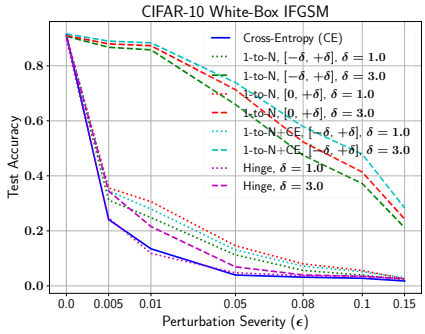
We are thankful to São Paulo Research Foundation (FAPESP grant #2017/12646-3) and National Council for Scientific and Technological Development (CNPq grant #309330/2018-1).

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Infor-*

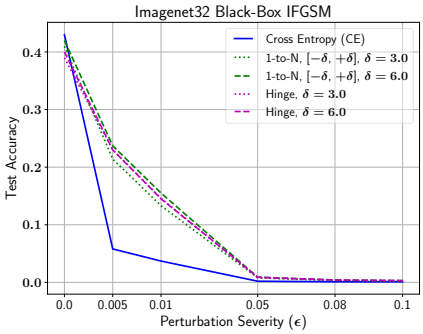


(a) IFGSM black-box attack

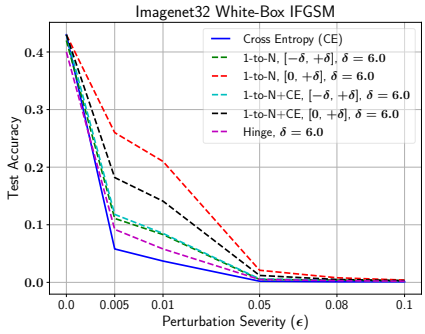


(b) IFGSM white-box attack

Fig. 7. Performance accuracy versus different values of perturbation for CIFAR-10 dataset.



(a) IFGSM black-box attack



(b) IFGSM white-box attack

Fig. 8. Performance accuracy versus different values of perturbation for ImageNet32 dataset.

mation Processing Systems, 2012, pp. 1097–1105.

[2] H. Zhang, C. Wu, Z. Zhang, Y. Zhu, Z. Zhang, H. Lin, Y. Sun, T. He, J. Mueller, R. Manmatha, M. Li, and A. Smola, "Resnest: Split-attention networks," *arXiv 2004.08955*, 2020.

[3] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *IEEE Computer Vision and Pattern Vision*. Computer Vision Foundation / IEEE, 2019, pp. 4690–4699.

[4] G. F. Elsayed, D. Krishnan, H. Mobahi, K. Regan, and S. Bengio, "Large margin deep networks for classification," in *32nd International Conference on Neural Information Processing Systems*, ser. NIPS18. Curran Associates Inc., 2018, p. 850860.

[5] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, Y. Bengio and Y. LeCun, Eds., 2015.

[6] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *International Conference on Learning Representations*, 2017.

[7] X. Liang, X. Wang, Z. Lei, S. Liao, and S. Z. Li, "Soft-margin softmax for deep classification," in *International Conference on Neural Information Processing*. Springer, 2017.

[8] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks," in *International Conference on Machine Learning*, 2016, pp. 507–516.

[9] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "Cosface: Large margin cosine loss for deep face recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2018, pp. 5265–5274.

[10] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2805–2824, 2019.

[11] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, no. 3, pp. 273–297, 1995.

[12] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*. MIT Press, 2018.

[13] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[14] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Computing Surveys*, 2020, https://doi.org/10.1145/3386252.

[15] J. Lin, C. Song, K. He, L. Wang, and J. E. Hopcroft, "Nesterov accelerated gradient and scale invariance for adversarial attacks." in *International Conference on Learning Representations*. OpenReview.net, 2020.

[16] Y. Tang, "Deep learning using support vector machines," *CoRR*, 2013, http://arxiv.org/abs/1306.0239.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, 1998.

[18] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009, https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[19] P. Chrabaszcz, I. Loshchilov, and F. Hutter, "A downsampled variant of imagenet as an alternative to the CIFAR datasets," *CoRR*, 2017.

[20] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *British Machine Vision Conference (BMVC)*, E. R. H. Richard C. Wilson and W. A. P. Smith, Eds., September 2016, pp. 87.1–87.12, https://dx.doi.org/10.5244/C.30.87.

[21] T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li, "Bag of tricks for image classification with convolutional neural networks," in *IEEE Computer Vision and Pattern Recognition*. Computer Vision Foundation / IEEE, 2019.