A DECIDABILITY-BASED LOSS FUNCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Computer vision problems often use deep learning models to extract features from images, also known as embeddings. Moreover, the loss function used during training strongly influences the quality of the generated embeddings. In this work, a loss function based on the decidability index is proposed to improve the quality of embeddings for the verification routine. Our proposal, the D-loss, avoids some Triplet-based loss disadvantages such as the use of hard samples and tricky parameter tuning, which can lead to slow convergence. The proposed approach is compared against the Softmax (cross-entropy), Triplets Soft-Hard, and the Multi Similarity losses in four different benchmarks: MNIST, Fashion-MNIST, CIFAR10 and CASIA-IrisV4. The achieved results show the efficacy of the proposal when compared to other popular metrics in the literature. The D-loss computation, besides being simple, non-parametric and easy to implement, favors both the inter-class and intra-class scenarios. Our code is available at: github link.

1 INTRODUCTION

Deep learning disrupted the computer vision field, especially regarding feature extraction and image representation. In this context, convolutional neural networks (CNNs) play a significant role (Krizhevsky et al., 2009; LeCun et al., 2015). Once deep CNNs are trained on a massive amount of data, it becomes a candidate to be used as a deep feature descriptor, not only in the domain or task in which the CNN was trained but in every computer vision problem, thanks to transfer learning techniques (Goodfellow et al., 2016). However, the loss function in which these CNNs were originally pre-trained, strongly influences the extracted deep representation features. The loss function is of paramount importance for training a CNN model and several approaches have been proposed in the literature (Sharif Razavian et al., 2014; Schroff et al., 2015; Parkhi et al., 2015; Wang et al., 2017; 2019a;c). Among them, one very popular is the cross-entropy loss (Goodfellow et al., 2016), also known as softmax-loss since it is often preceded by a softmax operation. Although originally designed for classification problems, it has been very successful in learning deep representative features along with CNN models (Krizhevsky et al., 2009; Sharif Razavian et al., 2014).

The softmax loss is widely used for several reasons, namely it is easy to interpret and implement, fast convergence, and for working well with different batch sizes (large or very small). Even though it is not designed for learning feature representations, the learned features are powerful enough for many tasks, such as face recognition (Parkhi et al., 2015; Wang et al., 2017), among others. However, in tasks in which it is necessary to know how close or how far the samples are concerning each other (such as the biometric verification task) on high-dimensional spaces, this type of loss is not the best option. There are efforts in the literature to adapt the softmax loss (Ranjan et al., 2017; Wang et al., 2017) to the task, but contrastive and triplet-based losses are still the ones that offer the best gains.

Contrastive loss better captures the relationship between two samples projected in a space (Euclidean, for example), by penalizing, during learning, negative samples (samples from other classes, also called here impostors) and rewarding samples from the same (samples from same class, also called here genuine) category (Chopra et al., 2005). Likewise, triplet-based loss explores the concept of similarities and dissimilarities between samples in a space, adding anchor elements. There are many triplet-based losses in the literature, such as triplet-center loss (He et al., 2018), quadruplet loss (Law et al., 2013) and in general, triplet-based loss produce better results, overcoming other pair-wise losses such N-pairs loss (Sohn, 2016), binomial deviance loss (Yi et al., 2014), histogram loss (Ustinova & Lempitsky, 2016) and Multi-Similarity Loss (Wang et al., 2019c).

Low convergence represents a major problem for triplet-based losses. Besides, given a set of samples, it is not trivial to find positive or negative instances to use as hard pairs, nor is it easy to fine-tune the margin that separates them (Parkhi et al., 2015). Notwithstanding, tripled-based losses are still the most popular losses in the literature, despite their limitations.

Contribution. In this paper, we propose a new loss function, the D-loss, based on the decidability index (Daugman, 2000). Daugman (2000) highlights this index as a quality measure for biometric systems, which is often used in the literature for this purpose (De Marsico et al., 2018; Luz et al., 2018). The D-loss assures both inter-class and intra-class separability, and, unlike triplet-loss, avoids the difficult problem of finding hard positive and negatives samples. It also provides better convergence, since it does not require parameter adjustment, and is easy to implement. The contributions of this paper can be summarized as (i) A new loss function, the D-Loss, based on the decidability index, which is intuitive and easy to compute and implement. The D-loss is also suitable for training models which aim at data representation and, unlike triplets loss, it favors both inter-class separability and intra-class approximation. (ii) Under the same conditions, the D-loss overcomes three other popular loss functions (Triplets Soft-hard Loss, Multi-Similarity Loss, and Softmax-Loss) in MNIST-FASHION, CIFAR-10 and presents comparable results for MNIST. Therefore, a competitive loss function. (iii) The D-loss has converged well on higher capacity networks such as the networks of the EfficientNet family. Results with D-loss overcome all the three other popular loss functions (Triplets Soft-hard Loss, Multi-Similarity Loss, and Softmax-Loss) evaluated here, for the biometric database CASIA-IrisV4 on the same model (B0) and training conditions.

2 Background and Related Works

Classification Losses - Softmax Loss. Given a training set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}, \mathbf{x}_i \in \mathbb{R}^{m \times n}$, and their respective labels $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_N\}, \mathbf{y}_i \in \{1, ..., C\}$, where C is the number of classes in the problem, the softmax loss function is given by:

$$\mathcal{L}_{Softmax} = -\sum_{i=1}^{N} \log \left(\frac{\mathrm{e}^{\left(\mathbf{w}_{y_i}^T f(\mathbf{x}_i) + b_{y_i}\right)}}{\sum_{j=1}^{C} \mathrm{e}^{\left(\mathbf{w}_j^T f(\mathbf{x}_i) + b_j\right)}} \right) \tag{1}$$

in which $\mathcal{L}_{Softmax} : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{K \times C}$ is the learned feature map, or embedding, K is the dimension of the embedding. The w_j and b_j, $j \in \{1, ..., C\}$, are the weights and biases of the last fully layer, respectively (see Figure 1 (a)). This formulation enforces features to have larger magnitudes and a radial distribution (Wang et al., 2017). Thus, it does not optimize the features to have small dissimilarity scores for positive pairs or higher dissimilarity scores for the negative ones. Other approaches tried to mitigate such effects on softmax-loss.

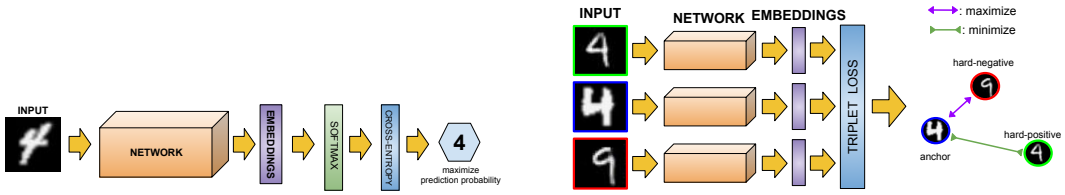


Figure 1: (a) Cross-entropy loss applied over a Softmax activation. To turn easier its reference in text, the name "Softmax loss" is used. (b) Triplet loss optimization schema. The same network is applied to all instances.

Pair-wise losses - Triplet losses. A triplet is a set of three components: an anchor x_i^a, a sample of the same class x_i^p (positive), and one from another class x_i^n (negative) (Parkhi et al., 2015; Schroff et al., 2015; Wang et al., 2019a). The positive and negative pairs share the same anchor, and the aim is to make the embedding f of the positive pair get closer, and the negative to separate according to

$$\|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha < \|f(x_i^a) - f(x_i^n)\|_2^2 \,, \tag{2}$$

in which α is a desired margin, and $\|.\|_2^2$ is the squared L2 distance. The triplet loss is given by

$$\mathcal{L}_{Triplet} = \sum_i^N \max\{0, \|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha\}, \tag{3}$$

for a set of N triplets. Nonzero values appear when the inequality equation 2 does not hold for a given margin α. Figure 1 (b) exhibits a sketch of the process, with the green arrow indicating the distance between a positive pair and in violet the distance between a negative. As in other pair-based losses, sampling is an important step (Wang et al., 2019c). The dataset must have sufficient recordings of the same class to use as positive pairs, otherwise the triplets are not viable. Moreover, randomly chosen x_i^p and x_i^n easily satisfies the condition equation 2 since the euclidean distances between the encodings of images of different people are likely to be large whilst those from the same person tends to be small. The main drawback of the random strategy is low convergence due to uninformative samples with negligible contributions (Wang et al., 2019c).

One solution is to perform hard mining by choosing x_i^p such that $\text{argmax}_{x_i^p} \|f(x_i^a) - f(x_i^p)\|_2^2$, and x_i^n with $\text{argmin}_{x_i^n} \|f(x_i^a) - f(x_i^n)\|_2^2$. This gives positive (negative) pairs whose distances to the anchor are the highest (smallest). In this case it is possible that $\|f(x_i^a) - f(x_i^n)\|_2^2 \approx \|f(x_i^a) - f(x_i^p)\|_2^2$ and the margin starts to play an important role.

The hard mining process demands high computational power and may lead to biased positive and negative pictures ranging from mislabeled to poorly imaged samples. Some strategies to overcome these issues are: (i) to generate triplets offline every n steps and compute argmin and argmax on a subset of the data; (ii) generate triplets online seeking hard pairs from the mini-batch; (iii) semi-hard negative samples (Schroff et al., 2015).

In some problems, images from the same class may present high dissimilarity when compared to images from different classes due to differences in lighting, color, and pose. The Ranked List Loss (RLL) (Wang et al., 2019a) and the Online Soft Mining (OSM) (Wang et al., 2019b) preserve intra-class data distribution instead of shrinking the encodings into a single point.

The Triplet-Center Loss (TCL) (He et al., 2018) replaces the hard negatives with the nearest negative centers from the center loss. Other mining strategies explore only moderate positive pairs (Shi et al., 2016), or consider different depths of similarity with sub-networks (Yuan et al., 2017).

Pair-wise losses - Multi-Similarity Loss. There are other pair-based loss functions, such N-pairs loss (Sohn, 2016), binomial deviance loss (Yi et al., 2014), histogram loss (Ustinova & Lempitsky, 2016) and Multi-Similarity Loss (Wang et al., 2019c). Among them, outstanding results have been reported with Multi-Similarity Loss. Therefore, we consider the Multi-Similarity Loss in this work for comparison purposes..

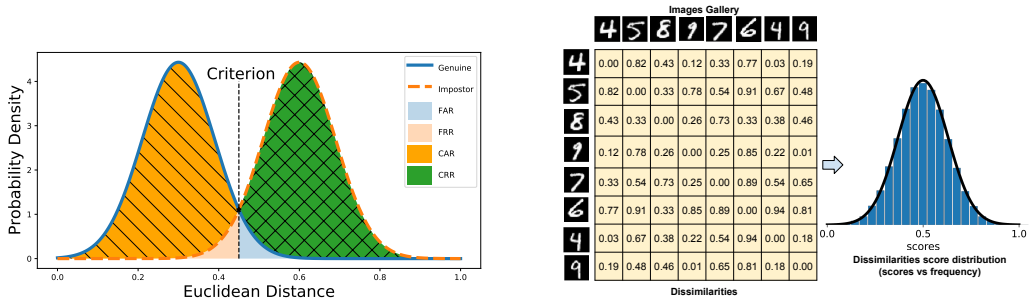
The Multi-Similarity Loss is based on the pair weighting formulation, which analyzes simultaneously three types of similarities before making a decision: self-similarity, negative relative similarity, and positive relative similarity. The approach consists of a two-step scheme: hard pairs selection (hard mining) and weighting. First, pairs are selected by means of positive relative similarity, then the selected pairs are weighted using both self-similarity and negative relative similarity, inspired by binomial deviance loss (Yi et al., 2014) and lifted structure loss (Oh Song et al., 2016). A framework is proposed to integrate the two steps, the General Pair Weighting (GPW) framework.

3 METHODOLOGY

We do not intend to compare our results with state-of-the-art metric learning methods, but rather to evaluate the robustness and discriminative potential of the representations obtained with the D-loss. We are especially interested in assessing D-loss performance in a biometric verification-like problem.

Decidability. A typical recognition system, such as biometric recognition, can be analyzed from four different perspectives: (i) False Accept Rate (FAR) in which an impostor is accepted as genuine, (ii) False Reject Rate (FRR) in which an genuine individual is classified as an impostor, (iii) Correct

Accept Rate (CAR) in which an genuine individual is accepted, and (iv) Correct Reject Rate (CRR) in which an impostor is correctly not accepted. The FAR and FRR are related to a system error and they are called Type I and Type II error respectively. It is worth mentioning that we discuss the distance function from a point of view of dissimilarity in this work. Figure 2 (a) shows the relation of the four perspectives when analyzed using the distribution of the scores.



(a) (b)

Figure 2: (a) Genuine and Impostor distributions. The Euclidean Distance is used to calculate the dissimilarity between two pairs. If the Euclidean Distance calculated is greater than a criterion, then, the individual is rejected, otherwise, it is accepted. (b) Process of construction of a Histogram based on images dissimilarities.

To illustrate, let X be a gallery of n samples. After propagating X through a neural network, one obtains $S = f(X)$, in which f is the embedding function, and S is a representation in the embedding space. The dissimilarity score curves (genuine and impostor) use S as input. The scores are generated by computing the distance of pairs of samples, on an all against all fashion, as shown in Figure 2 (b). The scores are further mapped into categories (or bins), and a histogram of the frequencies is generated by

$$F_i = \sum_j S_j \tag{4}$$

where F_i is the frequency count for a specific bin i, j the number of samples that meets the bin conditions and S_j the score related to the distance between a pair of samples.

Let P_G and P_I be the two distribution curves shown in Figure 2 (a) regarding genuine and impostor, respectively. The P_G is the probability density function of the dissimilarity scores, computed from the distance (through some distance metric) between two instances of the same class. Likewise, the P_I curve represents the probability density of the dissimilarity scores, computed from the distance between instances of different classes.

Thus, the four areas under the two distribution curves, such as illustrated in Figure 2 (a), express the probabilities of each decision metric (FAR, CAR, FRR, CRR).

The overlay of the authentic and the impostor scores is expected in a real scenario. The criterion threshold (C) can be manipulated and the FAR, FRR, CAR, and CRR updated according to users' needs. By manipulating this criterion and plotting the results in function of CAR and FAR, whose sum should result in one, a Receiver Operating Characteristic (ROC) or Neyman-Pearson curve can be created.

Ideally, the decision curve should be positioned as close as possible to the top left corner of the decision strategy curve in which the Correct Accept is close to 100% and the False Accept (error), close to 0%. However, this does not happen in the real world. To overcome those issues, arises the decidability index.

According to Daugman (2000), the decidability along with the distribution curves are good descriptors for the decision curve and can be better than FAR and FRR to assess pattern recognition systems performance.

The decidability can be defined as a correlation of genuine and impostor scores as defined in

$$Decidability = \frac{|\mu_I - \mu_G|}{\sqrt{\frac{1}{2}(\sigma_I^2 + \sigma_G^2)}}, \tag{5}$$

in which μ_G and μ_I are the means of the two distributions (histogram curve of genuine and impostor scores) and σ_I^2 and σ_G^2 are their corresponding standard deviations. The decidability indicates how far the genuine distribution scores curve is from the impostor and the overlap between these two curves (as shown in Figure 2 (a)).

Although the decidability index, as well as the other metrics presented in this subsection, is widely used in the biometric field, one could expand it to most pattern recognition problems. Since the decidability is independent of decision thresholds and quantifies, in a single scalar, the distance and overlap among two distribution curves, one hypothesis is that it might work as a loss function for training models aiming at image/signal representation.

D-loss. Many computer vision problems use deep learning models for data representation. Several authors use the cross-entropy loss (softmax-loss) to fine-tune models to a new domain or task, and it is also common to train models from scratch. To calculate the loss in a common supervised learning problem, with the cross-entropy loss, one uses the softmax operation to drive the output to probabilities (of each class), and the loss function aims to raise the accuracy on training data, according to a class hot encoded array. Thus, the problem is reduced to a classification problem, in with the main goal is not focused on the generation of better representation for the inputs, but on correctly classifying classes.

The state-of-the-art losses, triple-based ones, employ the concept of the anchor. Given an anchor sample, triplet-based loss aims to learn an embedding space, where positive pairs are forced to be closer to the anchor than the negative, by a margin. The pair-based losses, such as constrative (Chopra et al., 2005), triplet-center (He et al., 2018), and quadruplet (Law et al., 2013), put the focus on the generation of embeddings. Ideally, those pairs tend to bring more information and enhance the power of the model to represent the data. The pair selection represents a major issue for that kind of loss, since finding hard positive or negative samples related to anchor is not a trivial task. They also suffer from low convergence, and adjusting the margin parameter is not trivial (Parkhi et al., 2015).

We propose a new loss function, the D-loss. Differently from others, it makes use of all samples in a batch and does not rely on the concept of an anchor. The optimization of Eq. equation 5 separates μ_G and μ_I and reduces the variances σ_G and σ_I at the same time, thus improving both intra-class and inter-class scenarios as shown in Figure 3.

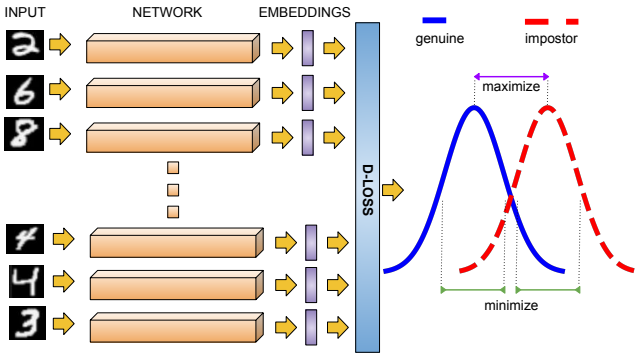


Figure 3: D-loss optimization schema.

The decidability index can rise to infinity, and the higher its value, the best is the separation between the impostor and genuine distributions. To better suit a minimization objective, we define D-loss as follows:

$$\mathcal{L}_{D\text{-}loss}(X, f) = \frac{1}{Decidability} \tag{6}$$

with $Decidability$ as defined in Eq. equation 5, and the embedding function f based on Euclidean Distance

$$d_{i,j} = ||f(x_i), f(x_j)||_2. \tag{7}$$

The X is given by $\{(x_i, y_i)\}_{i=1}^{N}$, in which N is the number of samples, x_i is the i^{th} training data and y_i is the class of the i^{th} sample. It is worth highlighting that the computation of the loss considers the entire batch, and the objective is to minimize the $\mathcal{L}_{D\text{-}loss}(X, f)$.

Training and Weights Updating. The training process using the D-loss is similar to training a traditional neural network. The main difference relies on the fact that the last layer of the CNN model is the embedding layer instead of the commonly used n neurons that represent the n classes of the problem. Still, the process of updating the weights is equal to the traditional, relying on a stochastic gradient descent optimization algorithm.

Several batches are created from the training data. Their size is a hyper-parameter of the network and the selection of the data within a batch is random. The weights are updated after the processing of each mini-batch. First, a distance is calculated for each pair within a mini-batch, and, based on those distances, the genuine and impostor distribution curves are computed. Subsequently, the mean and standard deviation of curves are calculated to obtain the decidability. The learning process aims to minimize decidability. The entire process is presented in Algorithm 1.

Algorithm 1: D-loss on one mini-batch.

Mini-Batch Setting: The batch size N, the number of classes C.
Input: $X = \{(x_i, y_i)\}_{i=1}^{N}$, the distance function f, the embedding function d, the learning rate β
Output: Updated f**.**
Step 1: Compute the embeddings by feeding-forward all images $\{x_i\}_{i=1}^{N}$.
Step 2: Compute the pair-wise distances.
    foreach $x_i \in X$ **do**
        foreach $x_j \in X$ **do**
            pdist[i, j] = d(x_i, x_j)
Step 3: Compute the genuine and impostor scores:
    foreach $x_i \in X$ **do**
        foreach $x_j \in X$ **do**
            if y_i == y_j *and* i != j **then**
                genuine.push(pdist[i, j])
            if y_i != y_j **then**
                impostor.push(pdist[i, j])
Step 4: Compute the decidability (Eq. equation 5).
Step 5: Compute $\mathcal{L}_{D\text{-}loss}(X, f)$ (Eq. equation 6).
Step 6: Gradient computation and back-propagation to update the parameters of the network.
    $\nabla_f = \frac{\partial \mathcal{L}_{D\text{-}loss}(X,f)}{\partial f}$ and $f = e - \beta \cdot \nabla_f$

4 EXPERIMENTS

Implementation Details. We implement the CNN functions and loss function with the TensorFlow/Keras Library. The CNN models are trained on a GPU GeForce Titan X with 12GB. In order to perform a fair comparison among losses and avoid experimental flaws such as pointed out by Musgrave et al. (2020), all losses are evaluated under the same conditions, fixing the network architecture, instances in the batches (same seed), optimization algorithm (Adam optimizer, initial learning rate of 10^{-3}), and same embedding dimension size.

Datasets. MNIST, Fashion-MNIST, and CIFAR-10 are used as proof of concept, mainly to investigate the convergence of D-loss in different domains. Also, these datasets are common benchmarks in machine learning. We also employ a popular biometric benchmark to evaluate the proposed loss in a verification scenario: CASIA-IrisV4. For a fair comparison, a 3-fold classification strategy is evaluated. All 249 individuals are split equally in three groups without overlap. All samples of an individual is just in one fold. To conduct the experiments, we set 30% of the train data for validation.

MNIST: The MNIST dataset of handwritten digits (LeCun et al., 2010) consists of gray-scale images with a size of 28x28 each. There are ten classes (zero to nine), and each contains 6,000 images for training and 1,000 for testing.

Fashion-MNIST: Similar to MNIST, the Fashion-MNIST (Xiao et al., 2017) has gray-scale images of 28x28 resolution, describing fashion items (shoes, coat, etc.). It is composed of 10 classes with the same MNIST distribution over the classes.

CIFAR-10: The CIFAR-10 (Krizhevsky et al., 2009) is a 10 class problem, with 6,000 images per class. It is a collection of 60,000 colored images, in which 10,000 images are for testing and 50,000 for training. Each image is a RGB of size 32x32.

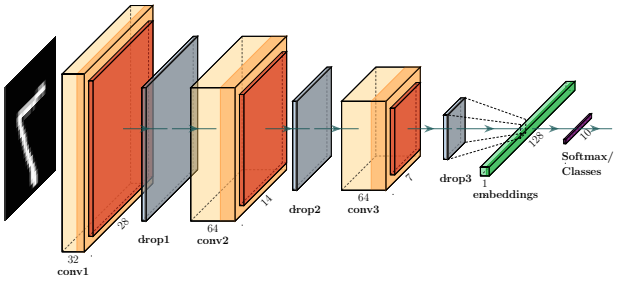
CASIA-IrisV4: The CASIA-IrisV4 contains six subsets and is an extension of CASIA-IrisV3 dataset. In this work, the CASIA-Iris-Interval (subset from CASIA-IrisV3) is used to report the results. The iris images are from 249 different subjects with a total of 1,438 JPEG images captured under near infrared illumination.

Evaluation Metrics. Metrics such as precision, recall, and accuracy are not the most suitable ones for biometric learning problems. Therefore, we use the False Acceptance Rate (FAR), False Rejection Rate (FRR), Equal Error Rate (EER), and Recall@K.

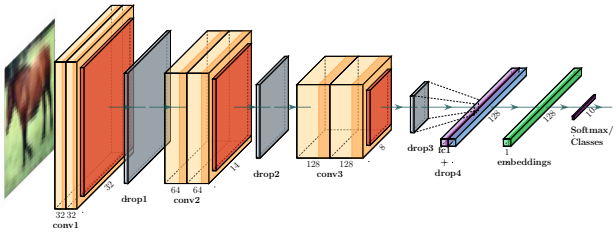
The EER is the point in which the FRR and FAR have the same values on the decision error trade-off curve. The FRR is the rate of incorrect rejections over different thresholds (zero to one), while the FAR is the rate of incorrect acceptances. The EER is the point of intersection of both FAR and FRR.

CNN Architecture and Model Training. We run our experiments on three different architectures. Two architectures are small networks (Figure 4) and each one matches the input size resolution of MNIST and CIFAR datasets (28×28 and 32×32). The architectures used with the MNIST, Fashion, and CIFAR-10 make use of simple operations that are well understood in the literature and are trained from scratch. The rationale to use such architectures (with simple convolutional blocks) is to emphasize the impact of the loss function.

An architecture with more capacity is selected for the CASIA-IrisV4 dataset. The experiments are run on the EfficientNet family network (B0 model) (Tan & Le, 2019). The architecture is larger, more sophisticated and has an input size resolution of 224×224.



(a) MNIST-architecture proposed in which *conv1*, *conv2* and *conv3* have filters size equal to 2x2, with *ReLU* activation and padding to avoid reduction after the convolution. All *poolings* are with the max function and window size of 2. *Dropout* of 30%.



(b) CIFAR-architecture proposed in which *conv1*, *conv2* and *conv3* have filters size equal to 3x3, with *ReLU* activation and padding to avoid reduction after the convolution. All *poolings* are with the max function and window size of 2. *Dropout* of 20%.

Figure 4: Architectures used during experiments.

All architectures have an embedding layer with dimension 256 (256-feature array output) with L2 normalization. The MNIST-architecture has 100,010 parameters, the CIFAR-architecture 567,082 parameters, and CASIA-V4-architecture 4,377,500 parameters. Before the loss layer, a Lambda Layer with an L2 normalization is used.

With these three architectures, we are able to evaluate the D-loss in different scenarios and from shallow and simple networks to one state-of-the-art network such as EfficientNet B0.

No data augmentation is implemented during training, and the euclidean distance is used to calculate the dissimilarity scores.

We employ Adam optimizer to train the models with a learning rate of 10^{-3}, and all images normalized to the range $[0, 1]$. The number of epochs is the same for all experiments, and dataset dependent: 100 epochs for the MNIST, 500 epochs for the Fashion-MNIST, $2,000$ epochs for CIFAR-10 and 1,000 epochs for CASIA-V4.

An initial experiment is performed on the MNIST dataset, to assess the impact of the batch size on both D-loss and Triplet-loss. We evaluate the batch sizes: 300, 400, and 500. As shown in Table 1, the batch size equal to 400 performed better on validation data for the D-loss. For triplet loss, results are very similar, both for batch sizes 300 and 400. Thus, a batch size of 400 is fixed for all losses to maintain the same evaluation setup in all experiments related to MNIST, Fashion and CIFAR-10. For the experiments on the CASIA-V4, a batch size of 150 is employed due to the network architecture size and hardware constraints.

Table 1: Batch size investigation on MNIST Dataset. Results on validation set (30% of train set).

Dataset	D-loss		Triplets Soft-hard Loss	
	EER (%)	Dec	EER (%)	Dec
300	1.07	7.92	0.70	7.63
400	0.47	9.31	0.74	7.25
500	1.01	8.22	0.83	7.10

Results and Literature Comparison. Tables 2 and 3 present a comparison between the D-loss, the Triplets soft-hard loss, the Multi Similarity Loss, and the Softmax-loss. The proposed approach outperformed the others in two out of four evaluated scenarios. On the remaining it exhibit comparable performance.

Table 2: Reported results in terms of Equal Error Rate (EER) in %. MS = Multi-Similarity; TS = Triplets Soft-hard; * = Model did not converge; * = No statistical significance.

Dataset	D-loss	MS Loss	TS Loss*	Softmax
MNIST (10)	1.04	1.06	**0.91**	1.50
Fashion (10)	**5.38**	5.82	5.94	7.58
CIFAR-10	**13.01**	14.11	20.01	14.08
CASIA-V4	**7.96 $\pm$ 3.43***	**7.84 $\pm$ 1.29***	38.55 $\pm$ 53.25	9.4 $\pm$ 1.92

Table 3: Reported results in terms of Recall@K (R@K). MS = Multi-Similarity; TS = Triplets Soft-hard; * = Model did not converged.

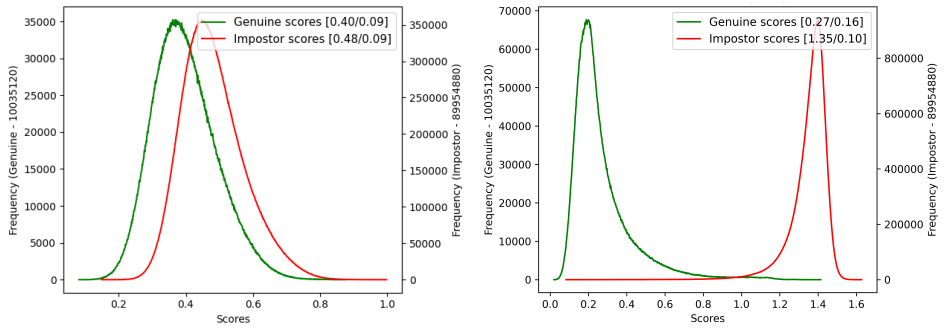
Dataset	R@K	D-loss	MS Loss	TS Loss*	Softmax
MNIST (10)	1	0.99	0.98	0.99	0.99
	2	0.99	0.99	0.99	1.00
	4	0.99	0.99	1.00	1.00
	8	0.99	1.00	1.00	1.00
Fashion (10)	1	0.88	0.88	0.89	0.89
	2	0.93	0.93	0.94	0.94
	4	0.96	0.96	0.96	0.96
	8	0.97	0.98	0.97	0.98
CIFAR-10	1	0.77	0.77	0.79	0.74
	2	0.84	0.84	0.85	0.82
	4	0.89	0.89	0.90	0.88
	8	0.92	0.92	0.93	0.92
CASIA-V4	1	0.99 $\pm$ 0.01	0.95 $\pm$ 0.03	0.66 $\pm$ 0.57	0.99 $\pm$ 0.01
	2	0.99 $\pm$ 0.00	0.98 $\pm$ 0.01	0.67 $\pm$ 0.58	0.99 $\pm$ 0.01
	4	1.00 $\pm$ 0.01	0.99 $\pm$ 0.01	0.67 $\pm$ 0.58	0.99 $\pm$ 0.01
	8	1.00 $\pm$ 0.01	1.00 $\pm$ 0.00	0.67 $\pm$ 0.58	1.00 $\pm$ 0.00

All scenarios present training overfitting as well. However, for the simple datasets (MNIST and Fashion-MNIST), a balanced degree of specialization and generalization is observed for all three losses.

Discussion. The main advantage of the D-loss over the triplets is that it does not require hard samples selection, which is a costly and difficult operation. That operation substantially increases the complexity of triplet-based losses. Using triplet-loss is also challenging when training a model from scratch Parkhi et al. (2015). As can be seen in Tables 2 and 3, the model did not converge properly for the CIFAR-10 and CASIA-V4 datasets.

A considerable disadvantage of the proposed loss is related to batch size. While the majority of the pair-based distance losses need an anchor and at least a negative sample, the proposed approach depends on a large number of both positive and negative samples to create the similarity (or dissimilarity) score distribution curves for the loss computation. Though regarding memory consumption, all pair-based losses analyzed here are comparable. While the D-loss and the triplets store distances between pairs, the multi-similarity loss stores the multiplication of the embeddings.

The impact of training with D-loss is more apparent in the genuine and impostor distributions curves, as one can see in Figure 5. Figure 5 (a) corresponds to the non-trained model, and Figure 5 (b) is the result after training. The distribution curves in Figure 5(?) indicate that the embeddings generated with the aid of d-loss are more robust.



(a) (b)

Figure 5: Distribution scores before (a) and after (b) training the model with D-loss on MNIST dataset. More distance between the mean of the curves implies better inter-class discrimination and the reduction in the width of the genuine distribution curve implies intra-class improvement.

5 Conclusion

Based on the decidability index, which is intuitive, easy to compute, and implement, we propose the D-loss as an alternative to triplet-based losses and the softmax-loss. The D-loss function is more suitable than softmax-loss for training models aiming to feature extraction / data representation, much like the Triplet-base loss. Moreover, the D-loss avoids some Triplet-based loss disadvantages, such as the use of hard samples, and it is non-parametric. Also, triplet-based losses have tricky parameter tuning, which can lead to slow convergence. The D-loss drawback is related to memory consumption since it requires a large batch size which can hinder the use of larger models.

The D-loss surpassed three other popular loss functions (Triplets Soft-hard Loss, Multi-Similarity Loss, and Softmax-Loss) in two out of three popular benchmark problems (MNIST-FASHION and CIFAR-10) and presented comparable results on a challenging scenario with the CASIA-IrisV4 dataset.

References

Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Conference on Computer Vision and Pattern Recognition*, volume 1, pp. 539–546, 2005.

John Daugman. Biometric decision landscapes. Technical report, University of Cambridge, Computer Laboratory, 2000.

Maria De Marsico, Michele Nappi, Fabio Narducci, and Hugo Proença. Insights into the results of MICHE I-mobile iris challenge evaluation. *Pattern Recognition*, pp. 286–304, 2018.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. Adaptive Computation and Machine Learning series. MIT Press, 2016.

Xinwei He, Yang Zhou, Zhichao Zhou, Song Bai, and Xiang Bai. Triplet-center loss for multi-view 3d object retrieval. In *Conference on Computer Vision and Pattern Recognition*, pp. 1945–1954. IEEE, 2018.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Marc T Law, Nicolas Thome, and Matthieu Cord. Quadruplet-wise image similarity learning. In *International Conference on Computer Vision*, pp. 249–256, 2013.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Eduardo Luz, Gladston Moreira, Luiz Antonio Zanlorensi Junior, and David Menotti. Deep periocular representation aiming video surveillance. *Pattern Recognition Letters*, pp. 2–12, 2018.

Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. In *European Conference on Computer Vision*, pp. 681–699. Springer, 2020.

Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *conference on computer vision and pattern recognition*, pp. 4004–4012, 2016.

Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *British Machine Vision Conference (BMVC)*, pp. 1–12. BMVA Press, 2015.

Rajeev Ranjan, Carlos D Castillo, and Rama Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv preprint arXiv:1703.09507*, 2017.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *conference on computer vision and pattern recognition*, pp. 815–823, 2015.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 806–813, 2014.

Hailin Shi, Yang Yang, Xiangyu Zhu, Shengcai Liao, Zhen Lei, Weishi Zheng, and Stan Z. Li. Embedding deep metric for person re-identification: A study against large variations. In *ECCV 2016 - Proceedings, Part I*, volume 9905 of *Lecture Notes in Computer Science*, pp. 732–748. Springer, 2016.

Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Neural Information Processing Systems*, pp. 1857–1865, 2016.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pp. 6105–6114. PMLR, 2019.

Evgeniya Ustinova and Victor Lempitsky. Learning deep embeddings with histogram loss. In *Neural Information Processing Systems*, pp. 4170–4178, 2016.

Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. In *Proceedings of the 25th ACM international conference on Multimedia*, pp. 1041–1049, 2017.

Xinshao Wang, Yang Hua, Elyor Kodirov, Guosheng Hu, Romain Garnier, and Neil M Robertson. Ranked list loss for deep metric learning. In *Conference on Computer Vision and Pattern Recognition*, pp. 5207–5216, 2019a.

Xinshao Wang, Yang Hua, Elyor Kodirov, Guosheng Hu, and Neil M. Robertson. Deep metric learning by online soft mining and class-aware attention. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):5361–5368, Jul. 2019b.

Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R Scott. Multi-similarity loss with general pair weighting for deep metric learning. In *Conference on Computer Vision and Pattern Recognition*, pp. 5022–5030, 2019c.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *Preprint arXiv:1708.07747*, 2017.

Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Deep metric learning for person re-identification. In *International Conference on Pattern Recognition*, pp. 34–39. IEEE, 2014.

Yuhui Yuan, Kuiyuan Yang, and Chao Zhang. Hard-Aware Deeply Cascaded Embedding. *International Conference on Computer Vision*, 2017-October:814–823, 2017. ISSN 15505499.