
Distinction Maximization Loss: Efficiently Improving Classification Accuracy, Uncertainty Estimation, and Out-of-Distribution Detection Simply Replacing the Loss and Calibrating

David Macêdo^{1,2}, Cleber Zanchettin¹, Teresa Ludermir¹

¹Centro de Informática, Universidade Federal de Pernambuco, Brasil

²Montreal Institute for Learning Algorithms, University of Montreal, Canada
d1m@cin.ufpe.br, cz@cin.ufpe.br, t1l@cin.ufpe.br

Abstract

Building robust deterministic deep neural networks is still a challenge. On the one hand, some approaches improve out-of-distribution detection at the cost of reducing classification accuracy in some situations. On the other hand, some methods simultaneously increase classification accuracy, out-of-distribution detection, and uncertainty estimation, but reduce inference efficiency, in addition to training the same model many times to tune hyperparameters. In this paper, we propose training deterministic deep neural networks using our DisMax loss, which works as a drop-in replacement for the commonly used SoftMax loss (i.e., the combination of the linear output layer, the SoftMax activation, and the cross-entropy loss). Starting from the IsoMax+ loss, we created novel logits that are based on the distance to all prototypes rather than just the one associated with the correct class. We also propose a novel way to augment images to construct what we call fractional probability regularization. Moreover, we propose a new score to perform out-of-distribution detection and a fast way to calibrate the network after training. Our experiments show that DisMax usually outperforms all current approaches simultaneously in classification accuracy, uncertainty estimation, inference efficiency, and out-of-distribution detection, avoiding hyperparameter tuning and repetitive model training. The code to replace the SoftMax loss with the DisMax loss and reproduce the results in this paper is available.¹

1 Introduction

Currently, deep neural networks have been used for classification in many applications. However, improving the robustness of such systems remains a significant challenge. Classification accuracy itself, out-of-distribution (OOD) detection performance, and uncertainty estimation comprise three major points regarding measuring the robustness of deep learning approaches.

On the one hand, most OOD detection approaches do not address uncertainty estimation or produce diminished classification accuracy in some cases [24, 7]. These solutions also require changing the training of the last layer by removing its weight decay to work properly. Therefore, they do not work as straightforward SoftMax loss drop-in replacements. On the other hand, some recent approaches that address both OOD detection and uncertainty estimation require hyperparameter tuning and/or reduce the efficiency of inferences when compared with pure deterministic deep neural networks [10, 25, 14]. Therefore, simultaneously increasing classification accuracy, OOD detection,

¹<https://github.com/dlmacedo/distinction-maximization-loss>

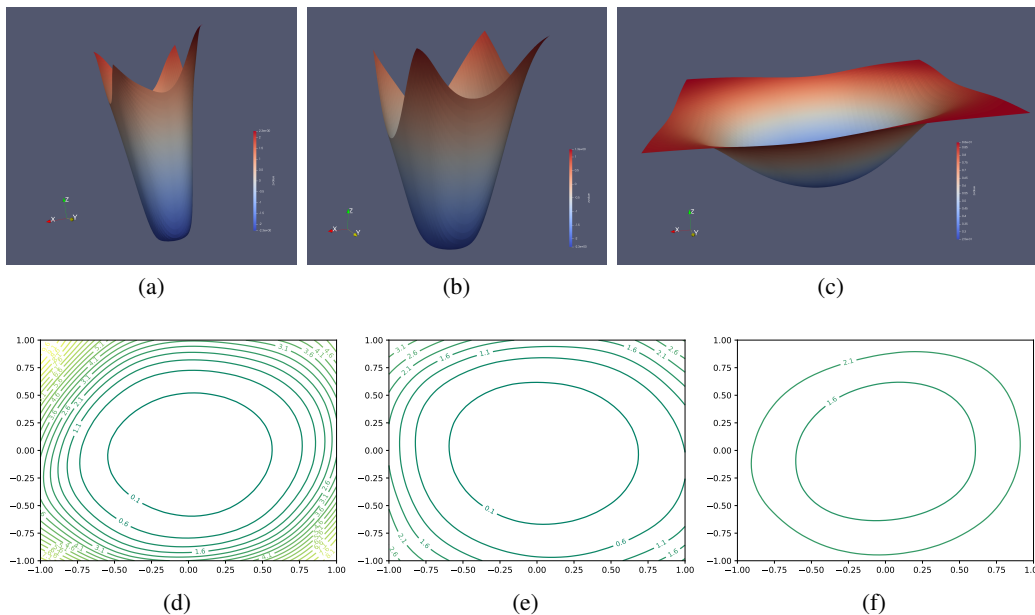


Figure 1: **Loss Surface Study.** 3D loss surfaces and 2D loss contours as proposed in [12] for ResNet34 trained on CIFAR10. (a, d) SoftMax; (b, e) IsoMax+; and (c, f) DisMax (ours). Considering that IsoMax+ usually outperforms SoftMax, and DisMax generally outperforms IsoMax+, we concluded that a less steep 3D inclination (i.e., a lower concentration of 2D contours) provides increased robustness performances.

and uncertainty estimation performances while maintaining inference efficiency poses a challenge, mainly if we also desire to avoid training the same architecture many times for hyperparameter tuning.

Recently, so-called IsoMax loss variants have been proposed [17, 16, 18]. They increase OOD detection performance without reducing inference efficiency compared with pure deterministic deep neural networks trained using the usual SoftMax loss (i.e., the combination of the linear output layer, the SoftMax activation, and the cross-entropy loss [15]). Moreover, they do not require repetitive neural network training for hyperparameter tuning. However, they neither increase the classification accuracy nor uncertainty estimation by themselves.

Contributions In this paper, starting from IsoMax+ loss [18], we construct the Distinction Maximization (DisMax) loss. Our main contributions are the following. First, we create the enhanced logits (logits+) by using all feature-prototype distances rather than just the feature-prototype distance to the correct class. Second, we introduce the fractional probability regularization (FPR) by minimizing the Kullback–Leibler (KL) divergence between the output probability distribution associated with an augmented image and a target probability distribution containing fractional rather than integer probabilities. We call DisMax dagger (DisMax^\dagger) the variant of our loss without using the FPR. Third, we construct a *composite* score for OOD detection that combines three components: the maximum logit+, the mean logit+, and the entropy of the network’s output. Fourth, we experimentally show that a simple temperature scaling after training makes DisMax produce high-performance uncertainty estimation. Similarly to the IsoMax and IsoMax+ losses, the DisMax loss works as a SoftMax loss drop-in replacement, as only a simple neural network training is required to use the proposed solution. Finally, we show experimentally that to obtain improved OOD detection performance, we need to construct losses with less steep 3D landscapes, as explained in Fig. 1.

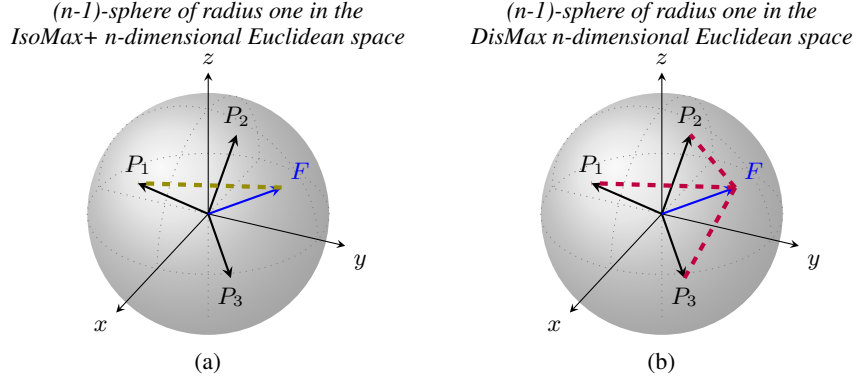


Figure 2: **All-Distances-Aware Logits, Enhanced Logits, or Logits+.** The illustration represents the difference between IsoMax+ [18] and DisMax with respect to logit formation. P_1 , P_2 , and P_3 represent prototypes of classes 1, 2, and 3, respectively. F denotes the feature associated with a given image. IsoMax+ constructs *each* logit associated with F considering its distance to a *single* prototype (olive dashed line). In contrast, DisMax loss builds *each* logit associated with F considering its distance to *all* prototypes (purple dashed lines). We use the terms all-distances-aware logits, enhanced logits, or logits+ terms indistinctly in this paper.

2 Distinction Maximization Loss

All-Distances-Aware Logits In IsoMax loss variants (IsoMax, IsoMax+, and DisMax), logits are formed from distances and are usually used to calculate the score to perform OOD detection. Hence, it is essential to build logits that contain semantic information relevant to separating in-distribution (ID) from OOD during inference. IsoMax+ uses the so-called isometric distances [18]. In the mentioned solution, the logits are simply the negatives of the isometric distances. We have two motivations to add the mean isometric distance (considering all prototypes) to the isometric distance associated with each class to construct what we call *all-distances-aware logits* or simply *enhanced logits* (logits+).

First, considering that IsoMax+ is an isotropic loss, the distances between the prototypes and ID are forced to become increasingly smaller. Therefore, after training, it is reasonable to believe that these distances are, on average, usually smaller than the distances from the prototypes to OOD, as they were not forced to be near the prototypes. OOD is not even used during training. Hence, adding their average to the distances used in IsoMax+ may help distinguish between ID and OOD more effectively. Second, taking all feature-prototype distances to compose the logits makes them a more stable source of information to perform OOD detection (Fig. 2).

$$\begin{aligned}
 &\text{all-distances-aware logit for the } j\text{-th class} \\
 &\text{mean isometric distance to all prototypes} \\
 L_+^j &= - \left(D_I^j + \frac{1}{N} \sum_{n=1}^N D_I^n \right) \tag{1} \\
 &\text{isometric distance to the } j\text{-th class prototype}
 \end{aligned}$$

$$\begin{aligned}
 &\text{predicted probability distribution} \\
 &\text{all-distances-aware logit for the } i\text{-th class} \\
 \mathcal{P}_{\text{DisMax}}(y^{(i)}|\mathbf{x}) &= \frac{\exp(L_+^i/T)}{\sum_{j=1}^N \exp(L_+^j/T)} \tag{2} \\
 &\text{all-distances-aware logit for the } j\text{-th class}
 \end{aligned}$$

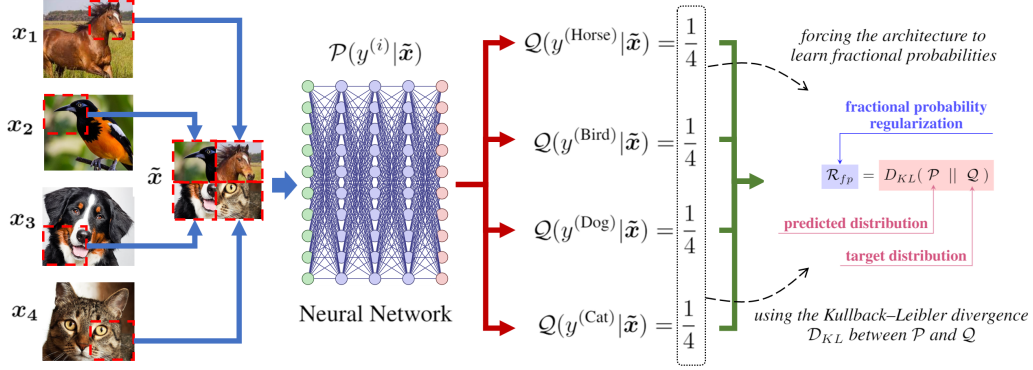


Figure 3: **Fractional Probability Regularization.** We use augmented images composed of patches of four randomly selected training examples. The KL divergence regularization term forces the network to predict *fractional* probabilities on augmented data.

Therefore, we consider an input \mathbf{x} and network that performs a parametrized transformation $\mathbf{f}_\theta(\mathbf{x})$. We also consider \mathbf{p}_ϕ^j to be the learnable prototype associated with class j . Moreover, considering that $\|v\|$ represents the 2-norm of a vector v , and \hat{v} represents 2-norm normalization of v , we can write the *isometric distance* relative to class j as $D_T^j = |d_s| \|\widehat{\mathbf{f}_\theta(\mathbf{x})} - \widehat{\mathbf{p}_\phi^j}\|$, where $|d_s|$ represents the absolute value of the *learnable* scalar called distance scale [18]. Finally, we can write the proposed enhanced logit for class j using the equation (1). The probabilities are given by the equation (2), where T is the temperature. For the rest of this paper, distance means *isometric* distance.

Fractional Probability Regularization Until now, we train neural networks with unitary probabilities. Indeed, the usual cross-entropy loss always forces a probability equal to one on a given training example. Currently, we force the neural network to learn by providing a miniscule proportion of points in the learning manifold. Hence, we propose what we call the *fractional probability regularization* (FPR). The idea is to force the network to learn a more diverse set of points in the learning manifold. Consequently, we confront the target and predicted probability distributions also on fractional probability values rather than only unitary probability manifold points. This procedure dramatically expands the points in the learning manifold from which the network can learn.

$$\begin{aligned}
 & \text{desired probability distribution} \\
 & \text{for the augmented image} \\
 & \mathcal{Q}_{\text{Target}}(y^{(i)}|\tilde{\mathbf{x}}) = \frac{1}{4} \sum_{m=1}^4 \delta[y^{(i)} - y^{(j^m)}] \quad (3) \\
 & \text{sum one quarter to the} \\
 & \text{probability of each class} \\
 & \text{in the augmented image} \\
 & \text{loss} \\
 & \text{function} \\
 & \mathcal{L}_{\text{DisMax}} = -\log^* \left(\frac{\exp(E_s L_+^k)}{\sum_j \exp(E_s L_+^j)} \right) + D_{KL}(\mathcal{P}_{\text{DisMax}}(y|\tilde{\mathbf{x}}) \parallel \mathcal{Q}_{\text{Target}}(y|\tilde{\mathbf{x}})) \quad (4) \\
 & \text{enhanced logit} \\
 & \text{for the correct class k} \\
 & \text{enhanced logit} \\
 & \text{for the j-th class} \\
 & \text{fractional probability} \\
 & \text{regularization}
 \end{aligned}$$

*The probability (i.e., the expression between the outermost parentheses) and logarithm operations are computed sequentially and separately for higher OOD detection performance [17].

Therefore, our batch is divided into two halves. In the first half, we use the usual unitary probability training. For the second batch, we construct augmented images specifically composed of patches of four others (Fig. 3). We construct our target probability distribution Q for those images by adding a quarter probability for each class corresponding to a patch of the augmented image. Finally, we minimize the KL divergence regularization between our predicted and target probability distribution on the second batch. These procedures do not increase training memory size requirements. Considering δ the Kronecker delta function, equation (3) represents the FPR in math terms. By combining the enhanced logits and the FPR, equation (4) presents the mathematical expression for the DisMax loss.

We recognize one similarity between CutMix [27] and FPR: both are based on the combination of images patches to create augmented data. However, we identify many differences. CutMix combines patches from two images, while FPR combines patches from four images. The format of the patches in CutMix is variable, as one of the images is usually not a rectangle, and the two patches typically have different areas. In FPR, all four patches always have the same size and format.

On the one hand, CutMix is randomly entirely applied to some batches and not to others. On the other hand, FPR is always applied to half of each batch. Unlike FPR, CutMix does not create a target probability distribution containing *fractional* probabilities and forces the predicted probability distribution to follow it by minimizing the KL divergence between them. Indeed, CutMix does not use the KL divergence at all. CutMix simply calculates the regular cross-entropy loss of the augmented image considering the labels of the original data and takes a linear interpolation between the resulting loss values. Therefore, while CutMix operates on loss values rather than probabilities, FPR operates directly on probabilities *before* calculating loss values. The concept of *fractional* probabilities is not even present in CutMix.

The design and theoretical differences mentioned above produce *significant practical consequences*. First, while CutMix *increases the training time*, FPR does not. Another relevant practical implication of the cited differences is the fact that CutMix *presents hyperparameters*. In addition to *adding a validation procedure* to the project pipeline, the need for reserving training data for validation to tune hyperparameters *may hurt the classification accuracy*, mainly on small datasets. Unlike CutMix, FPR does not introduce any hyperparameter. As FPR is an integral part of DisMax loss, the mentioned regularization is entirely transparent to the user of the provided code.

Max-Mean Logit Entropy Score For OOD detection, we propose a score composed of three parts. The first part is simply to calculate the maximum logit+. The second part is the mean logit+. Although each logit+ already considers the mean distances to all prototypes, we go a step further and now calculate the mean of the logits+ themselves. Finally, we subtract the entropy calculated considering the probabilities of the neural network’s output. We call this combined score *Max-Mean Logit Entropy Score* (MMLES), and it is given by equation (5). We call MMLES a *composite* score because it is formed by the sum of many other scores.

$$\begin{array}{c}
 \text{max-mean logit} \\
 \text{entropy score} \\
 \downarrow \\
 \mathcal{S}_{\text{MMLES}} = \max_j(L_+^j) + \frac{1}{N} \sum_{n=1}^N L_+^n - \mathcal{H}(\mathcal{P}_{\text{DisMax}}) \\
 \begin{array}{ccc}
 \uparrow & & \uparrow \\
 \text{maximum logit+} & & \text{mean logit+}
 \end{array}
 \end{array}
 \quad \begin{array}{c}
 \text{entropy} \\
 \downarrow \\
 \mathcal{H}(\mathcal{P}_{\text{DisMax}})
 \end{array}
 \quad (5)$$

Temperature Calibration Unlike the usual SoftMax loss, the IsoMax loss variants produce underestimated probabilities *to obey the Maximum Entropy Principle* [16, 18]. Therefore, we need to perform a temperature calibration after training to improve the uncertainty estimation. To find an optimal temperature, we used the L-BFGS-B algorithm with approximate gradients and bounds equal to 0.001 and 100 [1, 29, 21] for expected calibration error (ECE) minimization.

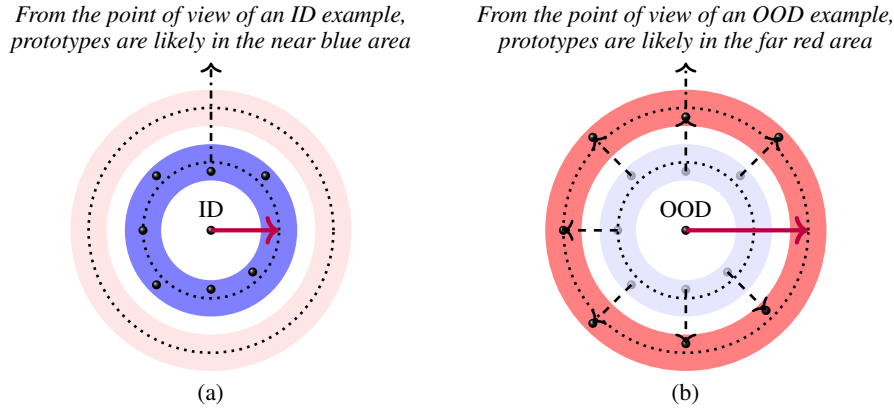


Figure 4: **Max-Mean Logit Entropy Score (MMLES)**. In addition to the maximum logit and the negative entropy, the MMLES incorporates the mean logit that considers the distances from the example feature vector to all prototypes of the class. The composition of many sources of information provides a high performance to MMLES. In DisMax, logits+ are used rather than usual logits.

3 Experiments

To allow standardized comparison, we used the datasets, training procedures, and metrics that were established in [6] and used in many subsequent OOD detection papers [13, 11, 5]. We trained many 100-layer DenseNetBCs with growth rate $k=12$ (i.e., 0.8M parameters) [8], 34-layer ResNets [4], and 28-layer WideResNets (widening factor $k=10$) [28] on the CIFAR10 [9] and CIFAR100 [9] datasets with SoftMax, IsoMax+, and DisMax losses using the same procedures (e.g., initial learning rate, learning rate schedule, weight decay). We used DisMax[†] for DenseNetBC trained on CIFAR10 because this is a very small model, and the mentioned dataset has too many examples per class; therefore, no augmentation is needed. For all other cases, we used DisMax.

We used stochastic gradient descent with the Nesterov moment equal to 0.9 with a batch size of 64 and an initial learning rate of 0.1. The weight decay was 0.0001, and we did not use dropout. We trained during 300 epochs. We used a learning rate decay rate equal to ten applied in epoch numbers 150, 200, and 250. Using DisMax, “you only *train* once”, as no hyperparameter tuning is required. We used resized images from TinyImageNet [2], the Large-scale Scene UNderstanding dataset (LSUN) [26], CIFAR10, CIFAR100, and SVHN [23] to create out-of-distribution samples. We added these out-of-distribution images to the validation sets of the in-distribution data to form the test sets and evaluate the OOD detection performance. We evaluated the accuracy (ACC) to assess classification performance.

We evaluated the OOD detection performance using the area under the receiver operating characteristic curve (AUROC), the area under the precision-recall curve (AUPR), and the true negative rate at a 95% true positive rate (TNR@TPR95). We used the expected calibration error (ECE) [22, 3, 20] for uncertainty estimation performance. The results are the mean and standard deviation of five runs. Two methods are considered to produce the same performance if their mean performance difference is smaller than the sum of the error margins.

Ablation Study The main aims of the ablation study are as follows: To verify whether the logits+ and FPR introduced by DisMax improve the accuracy and OOD detection performance, and to verify whether MMLES improves the OOD detection performance regarding MDS when using DisMax. As expected, Table 1 shows that logits+ and FPR often improve accuracy and OOD detection performance compared to IsoMax+ even when using MDS. Moreover, it also shows that replacing MDS with the *composite* score MMLES often increases OOD detection. These conclusions are essentially true regardless of the model, in-distribution, and (near, far, and very far) out-of-distribution.

Table 1: **Ablation Study.** MPS means Maximum Probability Score (i.e., the standard score for SoftMax loss). MDS means Minimum Distance Score (i.e., the standard score for IsoMax+ loss). MMLES means Max-Mean Logit Entropy Score (i.e., the standard score for DisMax loss). The best performances are bold.

CIFAR10								
Model	Method	Classification	Score	Out-of-Distribution Detection				
				Near		Far		Very Far
				CIFAR100	TinyImageNet	LSUN	SVHN	
		ACC (%) [↑]	MPS,MDS MMLES	TNR@95TPR (%) [↑]	TNR@95TPR (%) [↑]	TNR@95TPR (%) [↑]	TNR@95TPR (%) [↑]	
DenseNetBC100 (small size)	SoftMax (baseline) [6]	95.2±0.1	MPS	39.5±2.1	53.1±7.8	62.1±6.2	41.2±3.6	
	IsoMax+ [18]	95.1±0.1	MDS	57.3±1.1	86.9±0.4	91.4±0.3	96.4±0.6	
	DisMax [†] (ours)	95.1±0.1	MDS MMLES	56.2±0.5 54.2±1.0	88.0±0.3 89.0±1.0	92.2±0.3 92.4±1.1	97.5±0.3 98.3±0.3	
ResNet34 (medium size)	SoftMax (baseline) [6]	95.6±0.1	MPS	40.0±1.6	46.4±4.9	53.6±4.7	44.1±9.3	
	IsoMax+ [18]	95.5±0.1	MDS	55.1±1.3	71.0±6.4	81.5±4.4	82.4±8.8	
	DisMax (ours)	96.7±0.2	MDS MMLES	60.4±0.7 60.0±0.5	92.0±1.5 93.3±1.1	97.2±0.4 98.0±0.3	91.1±2.9 91.2±2.7	
WideResNet2810 (big size)	SoftMax (baseline) [6]	96.2±0.1	MPS	44.9±0.7	53.4±3.3	59.2±3.6	50.1±5.2	
	IsoMax+ [18]	96.0±0.1	MDS	61.5±0.4	80.2±4.2	87.4±3.0	96.3±1.4	
	DisMax (ours)	97.0±0.1	MDS MMLES	60.2±1.3 62.9±0.5	98.4±0.4 98.6±0.2	99.4±0.1 99.5±0.1	93.8±1.7 92.8±2.0	

CIFAR100								
Model	Method	Classification	Score	Out-of-Distribution Detection				
				Near		Far		Very Far
				CIFAR100	TinyImageNet	LSUN	SVHN	
		ACC (%) [↑]	MPS,MDS MMLES	TNR@95TPR (%) [↑]	TNR@95TPR (%) [↑]	TNR@95TPR (%) [↑]	TNR@95TPR (%) [↑]	
DenseNetBC100 (small size)	SoftMax (baseline) [6]	77.3±0.4	MPS	17.6±1.1	18.1±1.7	18.7±2.0	19.8±2.9	
	IsoMax+ [18]	76.9±0.3	MDS	17.2±0.7	71.6±6.5	66.8±9.4	67.1±3.0	
	DisMax (ours)	79.4±0.2	MDS MMLES	16.6±0.6 22.1±1.1	97.7±0.3 99.0±0.5	98.5±0.4 99.4±0.3	57.9±3.6 66.6±2.6	
ResNet34 (medium size)	SoftMax (baseline) [6]	77.7±0.3	MPS	19.4±0.5	20.6±2.4	21.3±3.4	17.1±5.0	
	IsoMax+ [18]	76.5±0.3	MDS	18.0±0.7	43.3±4.3	41.5±5.7	43.6±3.5	
	DisMax (ours)	80.6±0.3	MDS MMLES	20.8±0.4 22.0±0.5	79.9±1.5 85.4±1.7	81.5±1.4 86.4±1.3	43.7±1.6 48.5±2.0	
WideResNet2810 (big size)	SoftMax (baseline) [6]	79.9±0.2	MPS	21.8±0.7	26.7±5.9	28.7±6.7	15.8±5.5	
	IsoMax+ [18]	79.5±0.1	MDS	19.0±0.7	66.9±3.9	67.9±3.3	61.8±1.9	
	DisMax (ours)	83.0±0.1	MDS MMLES	22.4±0.2 24.6±0.3	92.3±1.3 96.3±1.2	95.2±0.4 97.8±0.9	56.8±1.8 65.6±1.2	

Classification, Efficiency, Uncertainty, and OOD Detection Results Table 2 compares DisMax with major approaches such as Scaled Cosine [24], GODIN [7], Deep Ensemble [10], DUQ [25], and SNGP [14] regarding classification accuracy, inference efficiency, uncertainty estimation, and (near, far, and very far) out-of-distribution detection. Unlike other approaches, DisMax is as inference efficient as a trivially trained neural network using the usual SoftMax loss. Moreover, using DisMax, “you only *train* once” the neural network, as no hyperparameter tuning is needed. Furthermore, DisMax often outperforms other approaches simultaneously in all evaluated metrics.

Additional Analyses Fig. 5 illustrates the distribution of mean logits+ under some scenarios. We see that prototypes are usually near to in-distribution than out-of-distribution, which explains why the mean all-distances-aware logit (i.e., a mean of means) improves OOD detection performance when combined with the maximum logit+ and the negative entropy to compose the MMLES.

Table 2: **Classification, Efficiency, Uncertainty, and OOD Detection Results.** Efficiency represents the inference speed (i.e., the inverse of the inference delay) calculated as a percentage of the performance of a single deterministic neural network trivially trained. For a fair comparison, we also calibrated the temperature of the SoftMax loss and IsoMax+ loss approaches using the same procedure that we defined for DisMax loss. Considering that input preprocessing can be applied indistinctly to improve the OOD detection performance of all methods compared [7] (at the cost of making their inferences approximately four times less efficient [16]), unless explicitly mentioned otherwise, all results are presented without using input preprocessing. The results worse than the baseline or most of the other approaches are shown in red. The methods that present the best performances are bold.

CIFAR10								
Model	Method	Classification	Inference	Uncertainty Estimation	Out-of-Distribution Detection			
					Near	Far	Very Far	
					CIFAR100	TinyImageNet	LSUN	SVHN
		ACC (%) [↑]	Efficiency (%) [↑]	ECE [↓]	AUPR (%) [↑]	AUROC (%) [↑]	AUROC (%) [↑]	AUPR (%) [↑]
DenseNetBC100 (small size)	SoftMax (baseline) [6]	95.2±0.1	100.0	0.0043±0.0008	86.2±0.5	92.9±1.6	94.7±0.9	93.7±3.3
	Scaled Cosine [24]	94.9±0.1	100.0	-	-	98.8±0.3	99.2±0.2	-
	GODIN+preprocessing ¹ [7]	95.0±0.1	26.0	-	-	99.1±0.1	99.4±0.1	-
	IsoMax+ [18]	95.1±0.1	100.0	0.0043±0.0012	90.4±0.3	97.6±0.9	98.3±0.5	99.7±0.1
	DisMax [†] (ours)	95.1±0.1	100.0	0.0045±0.0021	90.0±0.2	98.0±0.5	98.4±0.3	99.9±0.1
ResNet34 (medium size)	SoftMax (baseline) [6]	95.6±0.1	100.0	0.0060±0.0013	85.3±0.4	89.7±2.8	92.4±1.6	94.9±1.0
	GODIN [7]	95.1±0.1	100.0	-	-	95.6±0.5	97.6±0.2	-
	IsoMax+ [18]	95.5±0.1	100.0	0.0133±0.0177	90.1±0.3	95.1±1.0	96.9±0.6	98.7±0.6
	DisMax (ours)	96.7±0.2	100.0	0.0058±0.0008	90.3±0.2	98.3±0.3	99.5±0.1	99.1±0.3
WideResNet2810 (big size)	SoftMax (baseline) [6]	96.2±0.1	100.0	0.0038±0.0005	87.5±0.3	92.6±0.9	94.0±0.7	95.3±0.9
	Deep Ensemble [10]	96.6±0.1	10.3	0.0100±0.0010	88.8±1.0	-	-	96.4±1.0
	DUQ ² [25]	94.7±0.1	45.0	0.0340±0.0020	85.4±1.0	-	-	97.3±1.0
	SNGP ² [14]	95.9±0.1	62.5	0.0180±0.0010	90.5±1.0	-	-	99.0±1.0
	Scaled Cosine [24]	95.7±0.1	100.0	-	-	97.7±0.7	98.6±0.3	-
	IsoMax+ [18]	96.0±0.1	100.0	0.0107±0.0166	91.8±0.1	96.6±0.6	97.7±0.4	99.7±0.3
	DisMax (ours)	97.0±0.1	100.0	0.0043±0.0008	90.1±0.3	99.7±0.1	99.9±0.1	99.3±0.3

CIFAR100								
Model	Method	Classification	Inference	Uncertainty Estimation	Out-of-Distribution Detection			
					Near	Far	Very Far	
					CIFAR100	TinyImageNet	LSUN	SVHN
		ACC (%) [↑]	Efficiency (%) [↑]	ECE [↓]	AUPR (%) [↑]	AUROC (%) [↑]	AUROC (%) [↑]	AUPR (%) [↑]
DenseNetBC100 (small size)	SoftMax (baseline) [6]	77.3±0.4	100.0	0.0155±0.0026	71.3±0.8	71.8±2.2	73.1±2.4	87.5±1.5
	Scaled Cosine [24]	75.7±0.1	100.0	-	-	97.8±0.5	97.6±0.8	-
	GODIN+preprocessing ¹ [7]	75.9±0.1	24.0	-	-	98.6±0.2	98.7±0.0	-
	IsoMax+ [18]	76.9±0.3	100.0	0.0108±0.0017	71.3±0.4	95.1±1.1	94.2±1.7	97.4±0.6
	DisMax (ours)	79.4±0.2	100.0	0.0154±0.0006	74.4±0.2	99.8±0.1	99.9±0.1	96.4±0.8
ResNet34 (medium size)	SoftMax (baseline) [6]	77.7±0.3	100.0	0.0268±0.0015	73.3±0.1	79.0±2.1	79.6±1.7	86.3±3.3
	GODIN [7]	75.8±0.2	100.0	-	-	91.8±1.1	92.0±0.7	-
	GODIN+dropout ² [7]	77.2±0.1	100.0	-	-	87.0±1.1	87.0±2.2	-
	IsoMax+ [18]	76.5±0.3	100.0	0.0190±0.0025	72.1±0.4	89.7±1.0	89.8±1.3	94.5±0.6
	DisMax (ours)	80.6±0.3	100.0	0.0116±0.0014	74.2±0.6	97.6±0.5	97.7±0.6	94.8±1.0
WideResNet2810 (big size)	SoftMax (baseline) [6]	79.9±0.2	100.0	0.0272±0.0032	75.4±0.5	81.7±2.3	82.7±2.2	86.0±2.6
	Deep Ensemble [10]	80.2±0.1	12.3	0.0210±0.0040	78.0±1.0	-	-	88.8±1.0
	DUQ ² [25]	78.5±0.1	79.9	0.1190±0.0010	73.2±1.0	-	-	87.8±1.0
	SNGP ² [14]	79.9±0.1	74.9	0.0250±0.0120	80.1±1.0	-	-	92.3±1.0
	Scaled Cosine [24]	78.5±0.3	100.0	-	-	95.8±0.7	95.2±0.8	-
	IsoMax+ [18]	79.5±0.1	100.0	0.0188±0.0016	73.0±0.8	94.2±2.1	94.6±2.0	96.7±1.7
	DisMax (ours)	83.0±0.1	100.0	0.0143±0.0027	76.0±1.0	99.4±0.2	99.6±0.1	97.0±1.5

¹Requires reserving training data for validation and performing hyperparameter tuning.

²Requires training the same neural network many times for validating hyperparameters.

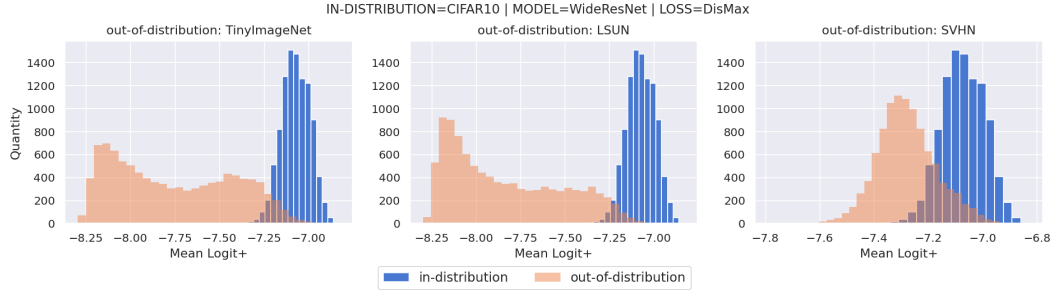


Figure 5: **Mean logit+ contribution to out-of-distribution detection.** In the feature space, the mean distance from an in-distribution image to *all* prototypes is usually smaller than the mean distance from an out-of-distribution image to the same prototypes. For example, consider a given class present in CIFAR10. This figure shows that even the prototypes associated with classes *other than the selected class* are usually nearer to images of the assumed class (in-distribution in blue) than images that do not belong to CIFAR10 at all (out-of-distributions in orange). This explains why the *mean value* of the logits+ considering *all prototypes* also contributes to the out-of-distribution detection performance. Hence, not only the distance to the nearest prototype is used in the mentioned task.

4 Related Works

In 2019, on the one hand, IsoMax [19] proposed a *end-to-end trainable* non-squared Euclidean distance last layer *to address out-of-distribution detection*. On the other hand, Scaled Cosine [24] proposed using a cosine distance. While the scaled factor in IsoMax is a *constant scalar* called the entropy scale, Scaled Cosine requires the addition of a *block of layers* to learn the scale factor. The mentioned block is composed of an exponential function, batch normalization, and a linear layer that has the features layer as input. Moreover, to present high performance, it is necessary to avoid applying weight decay to this *extra learning block*. We believe that this additional learning block, which adds an ad hoc linear layer in the final of the neural network, may make the solution prone to *overfitting* and explain the classification accuracy drop mentioned by the authors.

In 2020, GODIN [7] cited and was heavily inspired by Scaled Cosine. GODIN kept the *extra learning block* to learn the scale factor and also avoided applying weight decay to it. In addition to the usual affine transformation and cosine distance from Scaled Cosine, it presents a variant that uses a Euclidean distance-based last layer, similar to IsoMax. The major contribution of GODIN was to allow using the input preprocessing introduced in ODIN without the need for out-of-distribution data. However, input preprocessing increases the inference latency (i.e., reduces the inference efficiency) approximately four times [16]. Also in 2020, SNGP [14] cited and followed a path similar to IsoMax: *A distance-based output layer* for the detection of outside distribution. DUQ [25] also proposed a *distance-based loss function to address out-of-distribution detection*. Unlike IsoMax variants, in which we include DisMax, SNGP and DUQ are not as efficient as a single deterministic neural network. Moreover, they require training the neural network many times for hyperparameter tuning.

5 Conclusion

In this work, we proposed DisMax, a novel IsoMax loss variant that keeps working as a drop-in replacement for the SoftMax loss. We also presented a novel *composite* score called MMLES for OOD detection by combining the maximum logit+, the mean logit+, and the negative entropy of the network output. We present a simple temperature scaling procedure performed after training that makes DisMax produce high-performance uncertainty estimation.

Our experiments showed that the proposed method usually outperforms the current approaches simultaneously in classification accuracy, inference efficiency, uncertainty estimation, and out-of-distribution detection.

References

- [1] Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 1995.
- [2] Deng, J., Dong, W., Socher, R., Li, L., Li, K., and Li, F. ImageNet: A large-scale hierarchical image database. *Computer Vision and Pattern Recognition*, 2009.
- [3] Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. *International Conference on Machine Learning*, 2017.
- [4] He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. *European Conference on Computer Vision*, 2016.
- [5] Hein, M., Andriushchenko, M., and Bitterwolf, J. Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem. *Computer Vision and Pattern Recognition*, 2018.
- [6] Hendrycks, D. and Gimpel, K. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *International Conference on Learning Representations*, 2017.
- [7] Hsu, Y.-C., Shen, Y., Jin, H., and Kira, Z. Generalized ODIN: Detecting out-of-distribution image without learning from out-of-distribution data. *Computer Vision and Pattern Recognition*, 2020.
- [8] Huang, G., Liu, Z., Maaten, L. v. d., and Weinberger, K. Q. Densely connected convolutional networks. *Computer Vision and Pattern Recognition*, 2017.
- [9] Krizhevsky, A. Learning multiple layers of features from tiny images. *Science Department, University of Toronto*, 2009.
- [10] Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. *Neural Information Processing Systems*, 2017.
- [11] Lee, K., Lee, K., Lee, H., and Shin, J. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Neural Information Processing Systems*, 2018.
- [12] Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. Visualizing the loss landscape of neural nets. *Neural Information Processing Systems*, 2018.
- [13] Liang, S., Li, Y., and Srikant, R. Enhancing the reliability of out-of-distribution image detection in neural networks. *International Conference on Learning Representations*, 2018.
- [14] Liu, J. Z., Lin, Z., Padhy, S., Tran, D., Bedrax-Weiss, T., and Lakshminarayanan, B. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *Neural Information Processing Systems*, 2020.
- [15] Liu, W., Wen, Y., Yu, Z., and Yang, M. Large-margin softmax loss for convolutional neural networks. *International Conference on Machine Learning*, 2016.
- [16] Macêdo, D., Ren, T. I., Zanchettin, C., Oliveira, A. L. I., and Ludermir, T. B. Entropic out-of-distribution detection: Seamless detection of unknown examples. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
- [17] Macêdo, D., Ren, T. I., Zanchettin, C., Oliveira, A. L. I., and Ludermir, T. B. Entropic out-of-distribution detection. *International Joint Conference on Neural Networks*, 2021.
- [18] Macêdo, D. and Ludermir, T. Enhanced isotropy maximization loss: Seamless and high-performance out-of-distribution detection simply replacing the softmax loss. *CoRR*, abs/2105.14399, 2021.
- [19] Macêdo, D., Ren, T. I., Zanchettin, C., Oliveira, A. L. I., and Ludermir, T. Entropic out-of-distribution detection (first version). *CoRR*, abs/1908.05569v1, 2019.

- [20] Minderer, M., Djolonga, J., Romijnders, R., Hubis, F., Zhai, X., Houlsby, N., Tran, D., and Lucic, M. Revisiting the calibration of modern neural networks. *Neural Information Processing Systems*, 2021.
- [21] Morales, J. L. and Nocedal, J. Remark on “algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound constrained optimization”. *ACM Trans. Math. Softw.*, 2011.
- [22] Naeini, M. P., Cooper, G. F., and Hauskrecht, M. Obtaining well calibrated probabilities using bayesian binning. *AAAI Conference on Artificial Intelligence*, 2015.
- [23] Netzer, Y. and Wang, T. Reading digits in natural images with unsupervised feature learning. *Neural Information Processing Systems*, 2011.
- [24] Techapanurak, E., Suganuma, M., and Okatani, T. Hyperparameter-free out-of-distribution detection using cosine similarity. *Asian Conference on Computer Vision (ACCV)*, November 2020.
- [25] van Amersfoort, J. R., Smith, L., Teh, Y. W., and Gal, Y. Simple and scalable epistemic uncertainty estimation using a single deep deterministic neural network. *International Conference on Machine Learning*, 2020.
- [26] Yu, F., Zhang, Y., Song, S., Seff, A., and Xiao, J. LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, abs/1506.03365, 2015.
- [27] Yun, S., Han, D., Chun, S., Oh, S. J., Yoo, Y., and Choe, J. Cutmix: Regularization strategy to train strong classifiers with localizable features. *International Conference on Computer Vision*, 2019.
- [28] Zagoruyko, S. and Komodakis, N. Wide residual networks. *British Machine Vision Conference*, 2016.
- [29] Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Softw.*, 1997.