

Improving CNN Performance with Min-Max Objective

Weiwei Shi, Yihong Gong*, Jinjun Wang
 Institute of Artificial Intelligence and Robotics,
 Xi'an Jiaotong University, Xi'an 710049, China

Abstract

In this paper, we propose a novel method to improve object recognition accuracies of convolutional neural networks (CNNs) by embedding the proposed Min-Max objective into a high layer of the models during the training process. The Min-Max objective explicitly enforces the learned object feature maps to have the minimum compactness for each object manifold and the maximum margin between different object manifolds. The Min-Max objective can be universally applied to different CNN models with negligible additional computation cost. Experiments with shallow and deep models on four benchmark datasets including CIFAR-10, CIFAR-100, SVHN and MNIST demonstrate that CNN models trained with the Min-Max objective achieve remarkable performance improvements compared to the corresponding baseline models.

1 Introduction

Recent years have witnessed the bloom of convolutional neural networks (CNNs) in many computer vision and pattern recognition applications, including object recognition [Krizhevsky *et al.*, 2012], object detection [Girshick *et al.*, 2014], semantic segmentation [Girshick *et al.*, 2014], object tracking [Wang and Yeung, 2013], image retrieval [Hörster and Lienhart, 2008], etc. These great successes can be attributed to the following three main factors: (1) the rapid progress of modern computing technologies represented by GPGPUs and CPU clusters has allowed researchers to dramatically increase the scale and complexity of neural networks, and to train and run them within a reasonable time frame, (2) the availability of large-scale datasets with millions of labeled training samples has made it possible to train deep CNNs without a severe overfitting, and (3) the introduction of many training strategies, such as different types of activation functions [Krizhevsky *et al.*, 2012; Goodfellow *et al.*, 2013b; He *et al.*, 2015b; Jin *et al.*, 2015], different types of pooling [He *et al.*, 2014;

Gong *et al.*, 2014], dropout [Srivastava *et al.*, 2014], dropconnect [Wan *et al.*, 2013], pre-training [Dahl *et al.*, 2012], batch normalization [Ioffe and Szegedy, 2015], etc, has helped to generate better deep network models from the BP-based training process.

We argue that the strategies to improve object recognition accuracies by developing deeper, more complex network structures and employing ultra large scale training data are unsustainable, and are approaching their limits. This is because very deep models, such as GoogLeNet [Szegedy *et al.*, 2015], BN-Inception [Ioffe and Szegedy, 2015], ResNet [He *et al.*, 2015a] not only become very difficult to train and use, but also require large-scale CPU/GPU clusters and complex distributed computing platforms. These requirements are out of reach of many research groups with limited research budgets, as well as many real applications. If this trend continues, research and development in this field will soon become the privilege of a handful of large Internet companies.

The crux of object recognition is the invariant features. When an object undergoes identity-preserving image transformations, such as shift in position, change in illumination, shape, viewing angle, etc, the feature vector describing the object also changes accordingly. If we project these feature vectors into a high dimensional feature space (with the same dimension as that of the feature vectors), they form a low dimensional manifold in the space. The invariance is accomplished when each manifold belonging to a specific object category becomes compact, and the margin between two manifolds becomes large.

Based on the above observations, in this paper, we propose a novel objective, called Min-Max objective, to improve object recognition accuracies of CNN models. The Min-Max objective enforces the following properties for the features learned by a CNN model: (1) each object manifold is as compact as possible, and (2) the margin between two different object manifolds is as large as possible. The margin between two manifolds is defined as the Euclidian distance between the nearest neighbors of the two manifolds. In principle, the proposed Min-Max objective is independent of any CNN structures, and can be applied to any layers of a CNN model. Our experimental evaluations show that applying the Min-Max objective to the top layer is most effective for improving the model's object recognition accuracies.

To summarize, The main contributions of this paper are as

*Corresponding author: Yihong Gong(ygong@mail.xjtu.edu.cn)

follows:

- We propose the Min-Max objective to enforce the compactness of each manifold and the large margin between different manifolds for object features learned by a CNN model.
- Rather than directly modulating model weights and connecting structures as in typical deep learning algorithms, in our framework, the Min-Max objective directly modulates learned feature maps of the layer to which the Min-Max objective is applied.
- Experiments with different CNN models on four benchmark datasets demonstrate that CNN models trained with the Min-Max objective achieve remarkable performance improvements compared to the corresponding baseline models.

The remaining of this paper is organized as follows: Section 2 reviews related works. Section 3 describes our method, including the general framework, the formulation of the Min-Max objective and its detailed implementation. Section 4 presents the experimental evaluations and analysis. Section 5 provides discussions about the proposed method, and Section 6 concludes our work.

2 Related Work

Methods to improve object recognition accuracies of CNNs can be broadly divided into the following three categories: (1) increasing the model complexity, (2) increasing the training samples, and (3) improving training strategies. This section reviews representative works for each category.

Increasing the model complexity includes adding either the network depth or the number of feature maps in each layer.

Increasing the number of training samples often leads to improvement of recognition accuracies. Data augmentation [Krizhevsky *et al.*, 2012] is a low cost way of enlarging training samples by using label-preserving transformations, such as cropping and random horizontal flipping of patches, random scaling and rotation, etc. Another way to utilize more samples is to apply unsupervised learning [Wang and Gupta, 2015]. In practise, performance improvement by simply adding more training samples will gradually saturate. For example, it is reported that increasing the number of training samples from 1M to 14M from the ImageNet dataset only improved the image classification accuracy by 1%¹.

There are also methods that aim to improve training of CNN models to enhance performances. For instance, different types of activation functions, such as ReLU, LReLU, PReLU, APL, maxout, SReLU, etc, have been used to handle the gradient exploding and vanishing effect in BP algorithm. The dropout [Srivastava *et al.*, 2014] and dropconnect [Wan *et al.*, 2013] are proven to be effective at preventing neural networks from overfitting. Spatial pyramid pooling [He *et al.*, 2014] is used to eliminate the requirement that input images must be a fixed-size (e.g., 224×224) for CNN with fully connected layers. Batch normalization proposed by Ioffe and Szegedy [2015] can be used to accelerate deep

network training by reducing internal covariate shift. Lin *et al.* [2014] proposed the Network-In-Network (NIN) structure in which the convolution operation was substituted by a micro multi-layer perceptron (MLP) that slides over the input feature maps in a similar manner to the conventional convolution operation. The technique can be regarded as applying an additional 1×1 convolutional layer followed by ReLU activation, and therefore can be easily integrated into the current CNN pipelines. The Deeply-Supervised Nets (DSN) proposed by Lee *et al.* [2014] used a classifier, such as SVM or soft-max, at each layer during training to minimize both the final output classification error and the prediction error at each hidden layer.

3 The Proposed Method

3.1 General Framework

Let $\{\mathbf{X}_i, c_i\}_{i=1}^n$ be the set of input training data, where \mathbf{X}_i denotes the i^{th} raw input data, $c_i \in \{1, 2, \dots, C\}$ denotes the corresponding ground-truth label, C is the number of classes, and n is the number of training samples. The goal of training CNN is to learn filter weights and biases that minimize the classification error from the output layer. A recursive function for an M -layer CNN model can be defined as follows:

$$\mathbf{X}_i^{(m)} = f(\mathbf{W}^{(m)} * \mathbf{X}_i^{(m-1)} + \mathbf{b}^{(m)}), \quad (1)$$

$$i = 1, 2, \dots, n; m = 1, 2, \dots, M; \mathbf{X}_i^{(0)} = \mathbf{X}_i, \quad (2)$$

where, $\mathbf{W}^{(m)}$ denotes the filter weights of the m^{th} layer to be learned, $\mathbf{b}^{(m)}$ refers to the corresponding biases, $*$ denotes the convolution operation, $f(\cdot)$ is an element-wise non-linear activation function such as ReLU, and $\mathbf{X}_i^{(m)}$ represents the feature maps generated at layer m for sample \mathbf{X}_i . The total parameters of the CNN model can be denoted as $\mathbf{W} = \{\mathbf{W}^{(1)}, \dots, \mathbf{W}^{(M)}; \mathbf{b}^{(1)}, \dots, \mathbf{b}^{(M)}\}$ for simplicity.

As described in Section 1, we aim to improve object recognition accuracies of a CNN model by embedding the Min-Max objective into certain layer of the model during the training process. Embedding this objective into the k^{th} layer is equivalent to using the following cost function to train the model:

$$\min_{\mathbf{W}} L = \sum_{i=1}^n \ell(\mathbf{W}, \mathbf{X}_i, c_i) + \lambda \mathcal{L}(\mathcal{X}^{(k)}, \mathbf{c}), \quad (3)$$

where $\ell(\mathbf{W}, \mathbf{X}_i, c_i)$ is the classification error for sample \mathbf{X}_i , $\mathcal{L}(\mathcal{X}^{(k)}, \mathbf{c})$ denotes the Min-Max objective. The input to it includes $\mathcal{X}^{(k)} = \{\mathbf{X}_1^{(k)}, \dots, \mathbf{X}_n^{(k)}\}$ which denotes the set of produced feature maps at layer k for all the training samples, and $\mathbf{c} = \{c_i\}_{i=1}^n$ which is the set of corresponding labels. Parameter λ controls the balance between the classification error and the Min-Max objective.

Note that $\mathcal{X}^{(k)}$ depends on $\mathbf{W}^{(1)}, \dots, \mathbf{W}^{(k)}$. Hence directly constraining $\mathcal{X}^{(k)}$ will modulate the filter weights from 1^{th} to k^{th} layers (i.e. $\mathbf{W}^{(1)}, \dots, \mathbf{W}^{(k)}$) by feedback propagation during the training phase.

¹<http://image-net.org/challenges/LSVRC/2014/results>

3.2 Min-Max Objective

For $\mathcal{X}^{(k)} = \{\mathbf{X}_1^{(k)}, \dots, \mathbf{X}_n^{(k)}\}$, we denote by \mathbf{x}_i the column expansion of $\mathbf{X}_i^{(k)}$. The goal of the proposed Min-Max objective is to enforce both the compactness of each object manifold, and the max margin between different manifolds. Inspired by the Marginal Fisher Analysis research from [Yan *et al.*, 2007], we construct an intrinsic and a penalty graph to characterize the within-manifold compactness and the margin between the different manifolds, respectively, as shown in Figure 1. The intrinsic graph shows the node adjacency relationships for all the object manifolds, where each node is connected to its k_1 -nearest neighbors within the same manifold. Meanwhile, the penalty graph shows the between-manifold marginal node adjacency relationships, where the marginal node pairs from different manifolds are connected. The marginal node pairs of the c^{th} ($c \in \{1, 2, \dots, C\}$) manifold are the k_2 -nearest node pairs between manifold c and other manifolds.

Then, from the intrinsic graph, the within-manifold compactness can be characterized as:

$$\mathcal{L}_1 = \sum_{i,j=1}^n G_{ij}^{(I)} \|\mathbf{x}_i - \mathbf{x}_j\|^2, \quad (4)$$

$$G_{ij}^{(I)} = \begin{cases} 1, & \text{if } i \in \tau_{k_1}(j) \text{ or } j \in \tau_{k_1}(i), \\ 0, & \text{else,} \end{cases} \quad (5)$$

where $G_{ij}^{(I)}$ refers to element (i, j) of the intrinsic graph adjacency matrix $\mathbf{G}^{(I)} = (G_{ij}^{(I)})_{n \times n}$, and $\tau_{k_1}(i)$ indicates the index set of the k_1 -nearest neighbors of \mathbf{x}_i in the same manifold as \mathbf{x}_i .

From the penalty graph, the between-manifold margin can be characterized as:

$$\mathcal{L}_2 = \sum_{i,j=1}^n G_{ij}^{(P)} \|\mathbf{x}_i - \mathbf{x}_j\|^2, \quad (6)$$

$$G_{ij}^{(P)} = \begin{cases} 1, & \text{if } (i, j) \in \zeta_{k_2}(c_i) \text{ or } (i, j) \in \zeta_{k_2}(c_j), \\ 0, & \text{else,} \end{cases} \quad (7)$$

where $G_{ij}^{(P)}$ denotes element (i, j) of the penalty graph adjacency matrix $\mathbf{G}^{(P)} = (G_{ij}^{(P)})_{n \times n}$, $\zeta_{k_2}(c)$ is a set of index pairs that are the k_2 -nearest pairs among the set $\{(i, j) | i \in \pi_c, j \notin \pi_c\}$, and π_c denotes the index set of the samples belonging to the c^{th} manifold.

Based on the above descriptions, We propose the Min-Max objective which can be expressed as:

$$\mathcal{L} = \mathcal{L}_1 - \mathcal{L}_2. \quad (8)$$

Obviously, minimizing the Min-Max objective is equivalent to enforcing the learned features to form compact object manifolds and large margins between different manifolds simultaneously. The objective results in the following cost function:

$$\min_{\mathbf{W}} L(\mathbf{W}) = \sum_{i=1}^n \ell(\mathbf{W}, \mathbf{X}_i, c_i) + \lambda(\mathcal{L}_1 - \mathcal{L}_2), \quad (9)$$

In the next subsection, we will derive the optimization algorithm for Eq. (9).

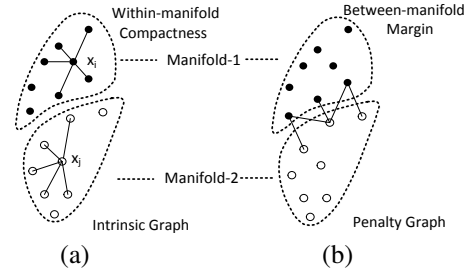


Figure 1: The adjacency relationships of (a) within-manifold intrinsic graph and (b) between-manifold penalty graph for the case of two manifolds. For clarity, the left intrinsic graph only includes the edges for one sample in each manifold.

3.3 Implementation

We use the back-propagation method to train the CNN model, which is carried out using mini-batch. Therefore, we need to calculate the gradients of the cost function with respect to the features of the corresponding layers.

Let $\mathbf{G} = (G_{ij})_{n \times n} = \mathbf{G}^{(I)} - \mathbf{G}^{(P)}$, then the Min-Max objective can be written as:

$$\mathcal{L} = \sum_{i,j=1}^n G_{ij} \|\mathbf{x}_i - \mathbf{x}_j\|^2 = 2tr(\mathbf{H}\Psi\mathbf{H}^T), \quad (10)$$

where $\mathbf{H} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$, $\Psi = \mathbf{D} - \mathbf{G}$, $\mathbf{D} = \text{diag}(d_{11}, \dots, d_{nn})$, $d_{ii} = \sum_{j=1, j \neq i}^n G_{ij}$, $i = 1, 2, \dots, n$, i.e. Ψ is the Laplacian matrix of \mathbf{G} , and $tr(\cdot)$ denotes the trace of a matrix.

The gradients of \mathcal{L} with respect to \mathbf{x}_i is

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}_i} = 2\mathbf{H}(\Psi + \Psi^T)_{(:,i)} = 4\mathbf{H}\Psi_{(:,i)}. \quad (11)$$

where, $\Psi_{(:,i)}$ denotes the i^{th} column of matrix Ψ .

To further improve the effectiveness of the proposed Min-Max objective, we also consider using the **kernel trick** to define the adjacency matrix, that is, $G_{ij}^{(I)}$ and $G_{ij}^{(P)}$ can be defined as:

$$G_{ij}^{(I)} = \begin{cases} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma^2}}, & \text{if } i \in \tau_{k_1}(j) \text{ or } j \in \tau_{k_1}(i), \\ 0, & \text{else,} \end{cases} \quad (12)$$

$$G_{ij}^{(P)} = \begin{cases} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma^2}}, & (i, j) \in \zeta_{k_2}(c_i), (i, j) \in \zeta_{k_2}(c_j) \\ 0, & \text{else.} \end{cases} \quad (13)$$

Then, the gradients $\frac{\partial \mathcal{L}}{\partial \mathbf{x}_i}$ for the kernel version of the Min-Max objective can be derived as follows.

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}_i} = 4\mathbf{H}(\Psi + \Phi)_{(:,i)}. \quad (14)$$

where, Φ denotes the Laplacian matrix of $\mathbf{V} = (V_{ij})_{n \times n}$, $V_{ij} = -\frac{G_{ij}}{\sigma^2} \|\mathbf{x}_i - \mathbf{x}_j\|^2$.

The total gradient with respect to \mathbf{x}_i is simply the combination of the gradient from the conventional CNN model and the above gradient from the Min-Max objective.

Table 1: Comparison results for the CIFAR-10 dataset.

Method	No. of Param.	Test Error(%)
Quick-CNN	0.145M	23.47
Quick-CNN+Min-Max	0.145M	18.06
Quick-CNN+kMin-Max	0.145M	17.59

Table 2: Comparison results for the CIFAR-100 dataset.

Method	No. of Param.	Test Error(%)
Quick-CNN	0.15M	55.87
Quick-CNN+Min-Max	0.15M	51.38
Quick-CNN+kMin-Max	0.15M	50.83

4 Experiments and Analysis

4.1 Overall Settings

To evaluate the effectiveness of the proposed Min-Max objective for improving object recognition accuracies of CNN models, we conduct experimental evaluations using shallow and deep models, respectively. Since our experiments show that embedding the proposed Min-Max objective to the top layer of a CNN model is most effective for improving its object recognition performance, in all experiments, we only embed the Min-Max objective into the top layer of the models and keep the other parts of networks unchanged. For the settings of those hyper parameters (such as the learning rate, weight decay, drop ratio, etc), we follow the published configurations of the original networks. All the models are implemented using the Caffe platform [Jia *et al.*, 2014] from scratch without pre-training. During the training phase, some parameters of the Min-Max objective need to be determined. For simplicity, we set $k_1 = 5$, $k_2 = 10$ for all the experiments, and it is possible that better results can be obtained by tuning k_1 and k_2 . σ^2 is empirically selected from $\{0.1, 0.5\}$, and $\lambda \in [10^{-6}, 10^{-9}]$.

4.2 Datasets

We conduct performance evaluations using four benchmark datasets, i.e. CIFAR-10, CIFAR-100, MNIST and SVHN. The reason for choosing these datasets is because they contain a large amount of small images (about 32×32 pixels), so that models can be trained using computers with moderate configurations within reasonable time frames. Because of this, the four datasets have become very popular choices for deep network performance evaluations in the computer vision and pattern recognition research communities.

CIFAR-10 Dataset. The CIFAR-10 dataset [Krizhevsky and Hinton, 2009] contains 10 classes of natural images, 50,000 for training and 10,000 for testing. Each image is a 32×32 RGB image.

CIFAR-100 Dataset. The CIFAR-100 dataset [Krizhevsky and Hinton, 2009] is the same in size and format as the CIFAR-10 dataset, except that it has 100 classes. The number of images per class is only one tenth of the CIFAR-10 dataset.

MNIST Dataset. The MNIST dataset [LeCun *et al.*, 1998] consists of hand-written digits 0-9 which are 28×28 gray images. There are 60,000 images for training and 10,000 images for testing.

Table 3: Comparison results for the SVHN dataset.

Method	No. of Param.	Test Error(%)
Quick-CNN	0.145M	8.92
Quick-CNN+Min-Max	0.145M	5.42
Quick-CNN+kMin-Max	0.145M	4.85

SVHN Dataset. The Street View House Numbers (SVHN) dataset [Netzer *et al.*, 2011] consists of 630,420 color images of 32×32 pixels in size, which are divided into the training set, testing set and an extra set with 73,257, 26,032 and 531,131 images, respectively. Multiple digits may exist in the same image, and the task of this dataset is to classify the digit located at the center of an image.

4.3 Experiments with Shallow Model

In this experiment, the CNN “quick” model from the Caffe package² (named Quick-CNN) is selected as the baseline model. This model consists of 3 convolution layers and 1 fully connected layers. Experimental results of test error rates on the CIFAR-10 test set are shown in Table 1. In this Table, Min-Max and kMin-Max correspond to the models trained with the Min-Max objective and the kernel version of the Min-Max objective respectively. From Table 1 it can be observed that, when no data augmentation is used, compared with the baseline model, the proposed Min-Max objective and its kernel version can remarkably reduce the test error rates by 5.41% and 5.88%, respectively.

We also evaluated the Quick-CNN model using the CIFAR-100 and SVHN datasets. The MNIST dataset can’t be used to test the model because images in the dataset are 28×28 in size, and the model only takes 32×32 images as its input. Table 2 and Table 3 list the respective comparison results. As can be seen, when no data augmentation is used, compared with the corresponding baseline model, the proposed Min-Max objective and its kernel version can remarkably reduce the test error rates by 4.49%, 5.04% respectively on CIFAR-100, and by 3.50%, 4.07% respectively on SVHN. The improvements on SVHN in terms of absolute percentage are not as large as those on the other two datasets, because the baseline Quick-CNN model already achieves a single-digit error rate of 8.92%. However, in terms of relative reductions of test error rates, the numbers have reached 39% and 46%, respectively, which are quite significant. These remarkable performance improvements demonstrate the effectiveness of the proposed Min-Max objective.

4.4 Experiments with Deep Model

Next, we apply the proposed Min-Max objective to the well-known NIN models [Lin *et al.*, 2014]. NIN consists of 9 convolution layers and no fully connected layer. Indeed, it is a very deep model, with 6 more convolution layers than that of the Quick-CNN model. Four benchmark datasets are used in the evaluation, including CIFAR-10, CIFAR-100, MNIST and SVHN.

The CIFAR-10 and CIFAR-100 datasets are preprocessed by the global contrast normalization and ZCA whitening as

²The model is available from Caffe package [Jia *et al.*, 2014]

Table 4: Comparison results for the CIFAR-10 dataset.

Method	No. of Param.	Test Error(%)
No Data Augmentation		
Stochastic Pooling	--	15.13
CNN + Spearmint	--	14.98
Maxout Networks	> 5M	11.68
Prob. Maxout	> 5M	11.35
NIN [Lin <i>et al.</i> , 2014]	0.97M	10.41
DSN	0.97M	9.78
NIN (Our baseline)	0.97M	10.20
NIN+Min-Max	0.97M	9.25
NIN+kMin-Max	0.97M	8.94
With Data Augmentation		
CNN + Spearmint	--	9.50
Prob. Maxout	> 5M	9.39
Maxout Networks	> 5M	9.38
DroptConnect	--	9.32
NIN [Lin <i>et al.</i> , 2014]	0.97M	8.81
DSN	0.97M	8.22
NIN (Our baseline)	0.97M	8.72
NIN+Min-Max	0.97M	7.46
NIN+kMin-Max	0.97M	7.06

Table 5: Comparison results for the CIFAR-100 dataset.

Method	No. of Param.	Test Error(%)
Learned Pooling	--	43.71
Stochastic Pooling	--	42.51
Maxout Networks	> 5M	38.57
Prob. Maxout	> 5M	38.14
Tree based priors	--	36.85
NIN [Lin <i>et al.</i> , 2014]	0.98M	35.68
DSN	0.98M	34.57
NIN (Our baseline)	0.98M	35.50
NIN+Min-Max	0.98M	33.58
NIN+kMin-Max	0.98M	33.12

in [Lin *et al.*, 2014; Lee *et al.*, 2014]. To be consistent with the previous works, for CIFAR-10, we also augmented the training data by zero-padding 4 pixels on each side, then corner-cropping and random horizontal flipping. In the test phase, no model averaging was applied, and we only cropped the center of a test sample for evaluation. No data whitening or data augmentation was applied for the MNIST dataset. For SVHN, we followed the training and testing protocols in [Lin *et al.*, 2014; Lee *et al.*, 2014], i.e., 400 samples per class selected from the training set and 200 samples per class from the extra set were used for validation, while the remaining 598,388 images of the training and the extra sets were used for training. The validation set was only used for tuning hyper-parameters and was not used for training the model. Preprocessing of the dataset again follows [Lin *et al.*, 2014; Lee *et al.*, 2014], and no data augmentation was used.

Table 4, 5, 6 and 7 show the evaluation results on the four benchmark datasets, respectively, in terms of test error rates. For fairness, for the baseline NIN model, we list the test results from both the original paper [Lin *et al.*, 2014] and our own experiments. In these tables, we also include the evaluation results of some representative methods, including Stochastic Pooling [Zeiler and Fergus,

Table 6: Comparison results for the MNIST dataset.

Method	No. of Param.	Test Error(%)
Multi-stage Architecture	--	0.53
Stochastic Pooling	--	0.47
NIN [Lin <i>et al.</i> , 2014]	0.35M	0.47
Maxout Networks	0.42M	0.47
DSN	0.35M	0.39
NIN (Our baseline)	0.35M	0.47
NIN+Min-Max	0.35M	0.32
NIN+kMin-Max	0.35M	0.30

Table 7: Comparison results for the SVHN dataset.

Method	No. of Param.	Test Error(%)
Stochastic Pooling	--	2.80
Maxout Networks	> 5M	2.47
Prob. Maxout	> 5M	2.39
NIN [Lin <i>et al.</i> , 2014]	1.98M	2.35
Multi-digit Recognition	> 5M	2.16
DropConnect	--	1.94
DSN	1.98M	1.92
NIN (Our baseline)	1.98M	2.55
NIN+Min-Max	1.98M	1.92
NIN+kMin-Max	1.98M	1.83

2013], CNN + Spearmint [Snoek *et al.*, 2012], Maxout Networks [Goodfellow *et al.*, 2013b], Prob. Maxout [Springenberg and Riedmiller, 2014], DroptConnect [Wan *et al.*, 2013], Learned Pooling [Malinowski and Fritz, 2013], Tree based priors [Srivastava and Salakhutdinov, 2013], Multi-stage Architecture [Jarrett *et al.*, 2009], Multi-digit Recognition [Goodfellow *et al.*, 2013a], and DSN [Lee *et al.*, 2014], which is the state-of-the-art model on these four datasets. It is worth mentioning that DSN is also based on the NIN structure with layer-wise supervisions.

From these tables, we can see that our proposed Min-Max objective achieves the best performance against all the compared methods. The evaluation results shown in the four tables can be summarized as follows.

- Compared with the corresponding baseline models, the proposed Min-Max objective and its kernel version can remarkably reduce the test error rates on the four benchmark datasets.
- On CIFAR-100, the improvement is most significant: NIN+kMin-Max achieved 33.12% test error rate which is 2.38% better than the baseline NIN model, and 1.45% better than the state-of-the-art DSN model.
- Although the test error rates are almost saturated on the MNIST and SVHN datasets, the models trained by the proposed Min-Max objectives still achieved noticeable improvements on the two datasets compared to the baseline NIN model and the state-of-the-art DSN model. In terms of relative reductions of test error rates, compared with the corresponding baseline models, the numbers have reached 36% and 28% respectively on these two datasets, which are significant.

These results once again substantiate the effectiveness of the proposed Min-Max objective.

4.5 Visualization

To get more insights of the proposed Min-Max objective, we visualize the top layer (i.e. the embedded layer) features obtained from two distinct models on the CIFAR-10 test set. The Quick-CNN and NIN models are selected as the respective baseline models. Figure 2 and Figure 3 show the feature visualizations for the two models respectively. It is clearly seen that, in both cases, the proposed Min-Max objective can help making the learned features with better within-manifold compactness and between-manifold separability as compared to the corresponding baseline models.

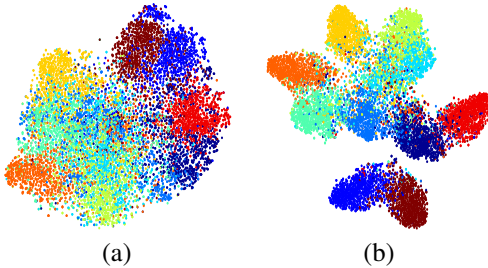


Figure 2: Feature visualization of the CIFAR10 test set by t-SNE [Van der Maaten and Hinton, 2008], using (a) Quick-CNN; (b) Quick-CNN+kMin-Max. A point denotes a test sample, and different colors represent different classes (The PCA produce quite similar visualization results to t-SNE).

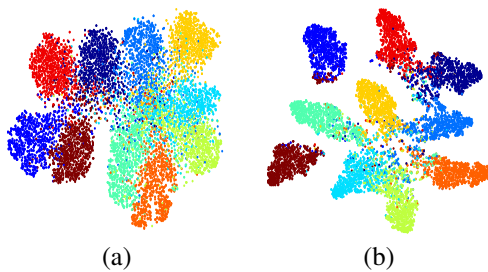


Figure 3: Feature visualization of the CIFAR-10 test set by t-SNE, using (a) NIN; (b) NIN+kMin-Max (The PCA produce quite similar visualization results to t-SNE).

4.6 Additional Computational Cost Analysis

CNN model is trained via stochastic gradient descent (SGD), where in each iteration the gradients can only be calculated from a mini-batch of training samples. In our method, during training phase, the additional computational cost is to calculate the gradients of the Min-Max objective with respect to the features of the embedded layer. According to Section 3.3, the gradients have the closed-form computational formula, which make the additional computation cost very low based on a mini-batch. In practise, the additional computational cost is negligible compared to that of the corresponding baseline CNN model.

Table 8: The effect of embedded position of the Min-Max objective on the performance on CIFAR-10 dataset with Quick-CNN model (i.e. Quick-CNN+Min-Max). *conv#* denotes a convolutional layer, *fc* denotes the fully connected layer.

Embedded Layer	<i>conv1</i>	<i>conv2</i>	<i>conv3</i>	<i>fc</i>
Test Error(%)	20.09	19.32	18.58	18.06

5 Discussions

To investigate the effect of the embedded position of the Min-Max objective on the performance, we conducted experiments with Quick-CNN on the CIFAR-10 dataset by embedding the Min-Max objective into different layers of the model, and the results of test error rates are shown in Table 8. We can clearly see that the higher the embedded layer, the better the performance. Interestingly, this concurs with the object recognition mechanism of the ventral stream in the human visual cortex. In recent years, research studies in the fields of physiology, biology, neuroscience, etc [DiCarlo *et al.*, 2012] have revealed that the human ventral stream consists of layered structures, where the lower layers strive to extract useful patterns or features that can well represent various types of objects/scenes in the natural environment, while higher layers serve to gradually increase the invariance degree of the low-level features to achieve accurate object/scene recognitions. These research discoveries suggest to us that it is more effective and appropriate to embed the invariance objectives into higher rather than lower layers of a deep CNN model. In the experiments, we also found that, for a given embedded layer, as long as the order of magnitude of λ is appropriate, the performance of our method does not change much, and the higher the embedded layer, the higher the order of magnitude of the optimal λ .

Moreover, it is clear from Figure 2(a) and 3(a) that through many layers of nonlinear transformations, a deeper network is able to learn good invariant features that can sufficiently separate objects of different classes in a high dimensional feature space. In contrast, features learned by a shallower network are not powerful enough to appropriately “untangle” those different object classes. This explains why the proposed Min-Max objectives are more effective for improving object recognition accuracies when applied to shallower networks.

6 Conclusion

In this paper, we propose a novel and general framework to improve the performance of CNN by embedding the proposed Min-Max objective into the training procedure. The Min-Max objective can explicitly enforce the learned object feature maps with better within-manifold compactness and between-manifold separability, and can be universally applied to different CNN models. Experiments with shallow and deep models on four benchmark datasets including CIFAR-10, CIFAR-100, SVHN and MNIST demonstrate that CNN models trained with the Min-Max objective achieve remarkable performance improvements compared to the corresponding baseline models. Experimental results substantiate the effectiveness of the proposed method.

Acknowledgments

This work is supported by National Basic Research Program of China (973 Program) under Grant No. 2015CB351705, the National Natural Science Foundation of China (NSFC) under Grant No. 61332018, and the Media Technology Laboratory, Central Research Institute, Huawei Technologies Company, Ltd.

References

- [Dahl *et al.*, 2012] G.E Dahl, D. Yu, L. Deng, and A. Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Trans. Audio, Speech, and Language Processing*, 2012.
- [DiCarlo *et al.*, 2012] J.J DiCarlo, D. Zoccolan, and N.C Rust. How does the brain solve visual object recognition? *Neuron*, 2012.
- [Girshick *et al.*, 2014] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.
- [Gong *et al.*, 2014] Y.C. Gong, L.W. Wang, R.Q. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. In *ECCV*. 2014.
- [Goodfellow *et al.*, 2013a] I.J Goodfellow, Y. Bulatov, J. Ibarz, S. Arnaud, and V. Shet. Multi-digit number recognition from street view imagery using deep convolutional neural networks. In *ICLR*, 2013.
- [Goodfellow *et al.*, 2013b] I.J. Goodfellow, D. Warde-Farley, M. Mirza, A.C. Courville, and Y. Bengio. Maxout networks. In *ICML*, 2013.
- [He *et al.*, 2014] K.M. He, X.Y. Zhang, S.Q. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014.
- [He *et al.*, 2015a] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.
- [He *et al.*, 2015b] K.M. He, X.Y. Zhang, S.Q. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, 2015.
- [Hörster and Lienhart, 2008] E. Hörster and R. Lienhart. Deep networks for image retrieval on large-scale databases. In *ACM MM*, 2008.
- [Ioffe and Szegedy, 2015] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.
- [Jarrett *et al.*, 2009] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *ICCV*, 2009.
- [Jia *et al.*, 2014] Y.Q. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*, 2014.
- [Jin *et al.*, 2015] X.J. Jin, C.Y. Xu, J.S. Feng, Y.C. Wei, J.J. Xiong, and S.C. Yan. Deep learning with s-shaped rectified linear activation units. In *AAAI*, 2015.
- [Krizhevsky and Hinton, 2009] A. Krizhevsky and G.E Hinton. Learning multiple layers of features from tiny images. *Master's thesis*, 2009.
- [Krizhevsky *et al.*, 2012] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.
- [LeCun *et al.*, 1998] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.
- [Lee *et al.*, 2014] C.Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *NIPS*, 2014.
- [Lin *et al.*, 2014] M. Lin, Q. Chen, and S. Yan. Network in network. In *ICLR*, 2014.
- [Malinowski and Fritz, 2013] M. Malinowski and M. Fritz. Learnable pooling regions for image classification. *CoRR*, 2013.
- [Netzer *et al.*, 2011] Y. Netzer, T. Wang, A. Coates, A. Bis-sacco, B. Wu, and A.Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS*, 2011.
- [Snoek *et al.*, 2012] J. Snoek, H. Larochelle, and R.P. Adams. Practical bayesian optimization of machine learning algorithms. In *NIPS*, 2012.
- [Springenberg and Riedmiller, 2014] J.T. Springenberg and M. Riedmiller. Improving deep neural networks with probabilistic maxout units. In *ICLR*, 2014.
- [Srivastava and Salakhutdinov, 2013] N. Srivastava and R.R Salakhutdinov. Discriminative transfer learning with tree-based priors. In *NIPS*, 2013.
- [Srivastava *et al.*, 2014] N. Srivastava, G.E Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 2014.
- [Szegedy *et al.*, 2015] C. Szegedy, W. Liu, Y.Q. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.
- [Van der Maaten and Hinton, 2008] L. Van der Maaten and G.E Hinton. Visualizing data using t-sne. *JMLR*, 2008.
- [Wan *et al.*, 2013] L. Wan, M. Zeiler, S. Zhang, Y. LeCun, and R. Fergus. Regularization of neural networks using dropconnect. In *ICML*, 2013.
- [Wang and Gupta, 2015] X.L Wang and A. Gupta. Unsupervised learning of visual representation using videos. In *ICCV*, 2015.
- [Wang and Yeung, 2013] Naiyan Wang and Dit-Yan Yeung. Learning a deep compact image representation for visual tracking. In *NIPS*, 2013.
- [Yan *et al.*, 2007] S.C. Yan, D. Xu, B. Zhang, H.J. Zhang, and Q. Yang ang S. Lin. Graph embedding and extensions: a general framework for dimensionality reduction. *IEEE Trans. PAMI*, 2007.
- [Zeiler and Fergus, 2013] M.D. Zeiler and R. Fergus. Stochastic pooling for regularization of deep convolutional neural networks. In *ICLR*, 2013.