Deep InterBoost networks for small-sample image classification

Li, Xiaoxu; Chang, Dongliang; Ma, Zhanyu; Tan, Zheng-Hua; Xue, Jing-Hao; Cao, Jie ; Jun, Goo

Published in:
Neurocomputing

DOI (link to publication from Publisher):
10.1016/j.neucom.2020.06.135

Publication date:
2021

Document Version
Accepted author manuscript, peer reviewed version

Deep InterBoost Networks for Small-sample Image Classification

Xiaoxu Li[a], Dongliang Chang[b], Zhanyu Ma[b,*], Zheng-Hua Tan[c], Jing-Hao Xue[d], Jie Cao[a], Jun Guo[b]

[a]*School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China*
[b]*Pattern Recognition and Intelligent System Laboratory, School of Artificial Intelligence, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[c]*Department of Electronic Systems, Aalborg University, Denmark*
[d]*Department of Statistical Science, University College London, London WC1E 6BT, U.K.*

Abstract

Deep neural networks have recently shown excellent performance on numerous image classification tasks. These networks often need to estimate a large number of parameters and require much training data. When the amount of training data is small, however, a network with high flexibility quickly overfits the training data, resulting in a large model variance and poor generalization. To address this problem, we propose a new, simple yet effective ensemble method called InterBoost for small-sample image classification. In the training phase, InterBoost first randomly generates two sets of complementary weights for training data, which are used for separately training two base networks of the same structure, and then the two sets of complementary weights are updated for refining the training of the networks through interaction between the two base networks previously trained. This interactive training process continues iteratively until a stop criterion is met. In the testing phase, the outputs of the two networks are combined to obtain one final score for classification. Experimental results on four small-sample datasets, UIUC-Sports, LabelMe, 15Scenes and Caltech101, demonstrate that the proposed ensemble method outperforms existing ones. Moreover, results from the Wilcoxon signed-rank tests show that our method is

*Corresponding author
Email address: mazhanyu@bupt.edu.cn (Zhanyu Ma)

statistically significantly better than the methods compared. Detailed analysis is also provided for an in-depth understanding of the proposed method.
Keywords: Ensemble learning, Deep neural network, Small-sample image classification, Overfitting.

1. Introduction

Image classification is an important application of machine learning and data mining [1, 2, 3]. Recent years have witnessed tremendous improvement in large-scale image classification due to the advances of deep learning [4, 5, 6, 7]. Despite the breakthroughs in applying deep networks, one persistent challenge is the classification with a small amount of training data [8, 9, 10, 11]. Recently, more and more studies focus on few-shot classification, in which each class contains 5, 10 or 20 labeled samples, for example. Apart from few-shot image classification, some studies also focus on the image classification that consists of hundreds of samples in each class. Here, we treat all these works as *small-sample image classification*. Small-sample image classification is important, not only because humans learn the visual concept of a class without the need of millions or billions of data, but also because many kinds of real-world data are small in quantity [12, 13].

Given limited training data points, a large network will easily encounter the overfitting problem [13, 14, 15]. There exist many methods aiming to alleviate the problem, such as dropout [16], large-margin losses [17, 18, 19], augmentation [20, 21, 22], fine-tuning [23, 24, 25] or weight decay [26]. However, when there are only a small number of data points in the training set, the overfitting problem will become inevitable [27]. This is mainly because a large network represents a large function space, in which many functions can fit a given small-sample dataset, making it difficult to find the underlying true function that is able to generalize well. As a result, a neural network trained with a small number of data points usually exhibits a large variance.

Ensemble learning is an effective way to reduce the model variance. Ac-

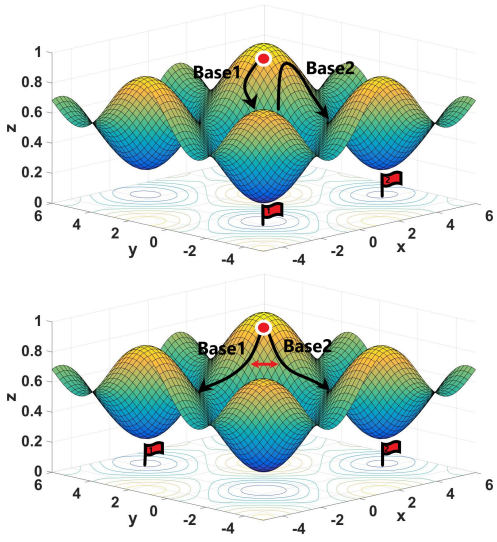

Figure 1: Motivation of InterBoost. In Snapshot (upper panel), the base networks (Base1 and Base2) are trained sequentially. In InterBoost (lower panel), the base networks (Base1 and Base2) are trained simultaneously and interactively. The interaction is indicated by a two-way red arrow. In each panel, the curved plane represents the loss function of base networks in Snapshot or InterBoost. The red flags indicate the local minima found by Snapshot or InterBoost.

cording to the error-ambiguity decomposition, which represents the ensemble generalization error as (the weighted average of) the generalization errors of the individual networks minus (the weighted average of) the ensemble ambiguities [28, 29], the variance can be reduced when multiple models or ensemble members are trained and combined for decision making, and the effect is more pronounced if ensemble members are with both accuracy and diversity [28, 30]. On the other hand, ensemble methods, including classical ones (Bagging, AdaBoost [31]) and new ensemble methods tailored to neural networks (Dropout [16], Snapshot Ensembling [32]) can increase either both the training and test costs of neural networks or one of them. With the increase of the training and test costs, the computational cost of ensembles of deep neural networks (DNN) can quickly become uneconomical and intolerable [32]. Hence, this paper only focuses on constructing an ensemble of a small number (e.g. two) of diverse and accurate base networks, to alleviate the big variance of DNN in small-sample image classification.

Among the existing ensemble methods for neural networks, Snapshot Ensembling is an effective method that introduces cyclic cosine annealing method in the training of neural networks to get multiple local minima of the loss function, and the network entities corresponding to these local minima are served as the ensemble members. In Snapshot Ensembling, the base networks are learned in sequence: the learned parameter values of a former base network are served as the initial parameters of the latter base network, see Fig. 1. *Inspired by Snapshot Ensembling, we also aim to find diverse local minima of the loss function but in a different way, and we propose a simple yet effective ensemble method called InterBoost, in which base networks are trained simultaneously and interactively, see Fig. 1.* Specifically, in our proposed InterBoost the original data are first re-weighted by two sets of complementary weights. Secondly, the two base neural networks with the same structure are separately trained by minimizing the two re-weighted loss functions. Then we update weights of training data according to the prediction scores of the two base networks on the training data, in a way that the weight of a data point for one base network is increased if the

prediction score of the base network is lower than the other for that data point. As a result, we obtain an ensemble that can increase gradually the accuracies of base networks while encouraging diversity between base networks.

We present the training and test procedures of the proposed InterBoost, and evaluate it on four small-sample datasets (UIUC-Sports [33], LabelMe [34], 15Scenes [35], and Caltech101 [36]) with a comparison to Bagging, AdaBoost, mixture of experts, Snapshot Ensembling and some other existing methods. Experimental results show the superior performance of the proposed InterBoost. Results from the Wilcoxon signed-rank tests [37, 38] also show that InterBoost is statistically significantly better than the other methods.

2. Related work

In the study of neural networks, ensemble methods can be roughly classified into the following groups.

Bagging and its variants. The strategy of Bagging trains the base classifiers on the bootstrap samples generated from the training dataset and then combines the classifiers based on some rules, e.g. a weighted average [31, 39]. The Bagging methods attempt to obtain the diversity from the bootstrap sampling, i.e. random sampling with replacement. This sampling approach makes the base classifiers in Bagging have a large generalization error on small-sample data although the diversity among base classifiers is achieved.

Boosting and its variants. The strategy of Boosting starts from a classifier trained on all the available training data and then sequentially trains the new member classifiers [40, 41, 42]. Taking AdaBoost [31] as an example: a new member classifier is trained on a re-weighted dataset, in which the re-weighting is based on the training errors generated from the previous classifiers. Therefore, AdaBoost works well for the weak base classifiers. However, if the base classifier is of high complexity, such as a large scale neural network, it may have no training error on the training samples. Consequently, the second classifier will

be trained on the original training data from the scratch again. In this case, the diversity of base networks in AdaBoost is mainly left to the randomness obtained from the initialization of the network parameters and the stochastic gradient descent (SGD) [43, 44], rather than from the re-weighting as usual.

Mixture of experts. The mixture of experts (MoE) [45, 46, 47] is also an effective approach to exploiting multiple learners, namely the experts [31]. MoE works in a divide-and-conquer strategy, where a complex task is broken up into several simpler and smaller subtasks. Next, the individual learners are trained for different subtasks. In particular, taking an individual learner which is a neural network as an example: an MoE network can be composed of a gating network and multiple subnetworks. The gating network is usually employed to combine the experts, and the subnetworks will focus on the subregions of the solution space for "subtasks" [48, 49]. Implicitly, each subnetwork is trained primarily by a subset of the training data, which may present a limitation for small-sample classification problems.

New ensembles for neural networks. Apart from the classical ensemble methods mentioned above, there also exist some new ensemble methods in the area of neural networks, e.g. Dropout [16], DropConnect [50] and Stochastic Depth techniques [51], which create an ensemble by dropping some hidden nodes, connections (weights) and layers, respectively. There also exist some ensemble methods taking advantage of characteristics of neural networks, such as Boosted Residual Networks, Snapshot Ensembling and Temporal ensembling. In [52], Boosted Residual Networks are generated by increasing the size of an original residual network via adding one residual block at each round of boosting. Snapshot Ensembling [32] is a method able to, by training only one time and finding multiple local minima of an objective function, get many ensemble members. Temporal ensembling [53], a parallel work to Snapshot Ensembling, trains a single network and uses the predictions made on different epochs as an ensemble prediction of multiple subnetworks because of the dropout regularization.

3. The proposed InterBoost method

3.1. Initialization of sample weights

For a training set $\{\boldsymbol{x}_d, \boldsymbol{y}_d\}$, $d \in \{1, \ldots, D\}$, where $\boldsymbol{y}_d$, a one-hot variable, is the true class label of $\boldsymbol{x}_d$, we assign weights to data points $\{\boldsymbol{x}_d, \boldsymbol{y}_d\}$, which are used for re-weighting the loss of the points in the loss function of a neural network. It is equivalent to changing the distribution of the training data and thus changing the optimization objective of the neural network. We randomly assign weights $W_{1d} \in (0, 1)$ to $\{\boldsymbol{x}_d, \boldsymbol{y}_d\}$ for training the first base network, and then assign complementary weights $W_{2d} = 1 - W_{1d}$ to $\{\boldsymbol{x}_d, \boldsymbol{y}_d\}$ for training the second base network.

3.2. InterBoost training

The core idea of the proposed InterBoost is to train two base neural networks interactively. This is in contrast to Boosting, where base networks are typically trained in sequence, namely the subsequent network or learner is trained on a dataset with new data weights that are updated using the error rate performance of the previous base network. As there is an interaction between the two base networks during the training process, we call the proposed method *InterBoost*, the training procedure of which is shown in Fig. 2.

The training procedure of InterBoost contains N iterations. In the nth iteration ($n \in \{1, \ldots, N\}$), it first trains the ith base network ($i \in \{1, 2\}$) using re-weighted training data $\{\boldsymbol{x}_d, \boldsymbol{y}_d, W_{id}^{(n)}\}$, $d \in \{1, \ldots, D\}$, and then interactively updates data weights based on the probability that the ith base network classifies $\boldsymbol{x}_d$ correctly, namely $P(\boldsymbol{y}_d|\boldsymbol{x}_d, \boldsymbol{\theta}_i^{(n)})$, where $\boldsymbol{\theta}_i$ represents the parameters of the ith base network. Here, we suppose that the activation function of the last layer in the ith base networks is *Softmax*, $P(\boldsymbol{y}_d|\boldsymbol{x}_d, \boldsymbol{\theta}_i^{(n)})$ is obtained by computing the dot product of $\boldsymbol{y}_d$ and the output vector of the ith base network with input $\boldsymbol{x}_d$. Training networks and updating data weights run alternately until a stop condition is met.

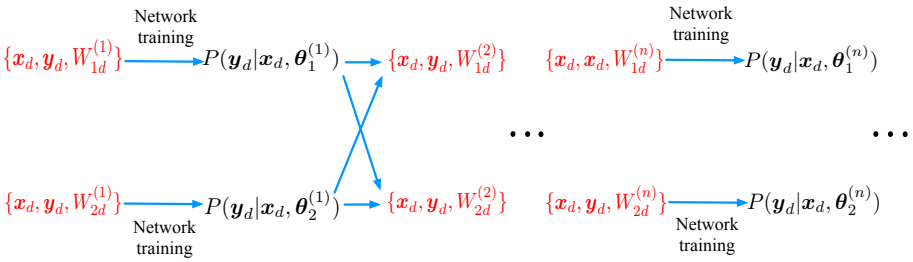

Figure 2: Illustration of the training procedure of InterBoost. In the plot, W_{1d} and W_{2d} are the weights of data point $\{\boldsymbol{x}_d, \boldsymbol{y}_d\}$, $d \in \{1, 2, \ldots, D\}$, for the two base networks, respectively; $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ are the parameters of two base networks; $W_{1d}^{(n)} + W_{2d}^{(n)} = 1$ with $W_{1d}^{(n)} \in (0, 1)$, $W_{2d}^{(n)} \in (0, 1)$ and n is the number of iteration; $P(\boldsymbol{y}_d|\boldsymbol{x}_d, \boldsymbol{\theta}_i^{(n)})$, $i \in \{1, 2\}$ is the probability that the ith base network can classify $\boldsymbol{x}_d$ into $\boldsymbol{y}_d$th class after the nth iteration.

Updating of $\boldsymbol{\theta}_i$. To compute $\boldsymbol{\theta}_i^{(n)}$, $i \in \{1, 2\}$ in the nth iteration, we minimize the weighted cross entropy loss functions $L_i^{(n)}$, with $L_i^{(n)}$ expressed as

$$L_i^{(n)} = -\sum_{d=1}^{D} W_{id}^{(n)} \log(P(\boldsymbol{y}_d|\boldsymbol{x}_d, \boldsymbol{\theta}_i^{(n-1)})). \tag{1}$$

Updating of W_{id}. For W_{id}, $i \in \{1, 2\}$, we devised the following updating rule: If the prediction of a data point in one base network is higher than that in another, its weight in next iteration for training this base network will be smaller than its weight for training another base network. In this way, a base network will be assigned a larger weight for a data point on which it does not perform well. Hence the interaction makes each base network focus on diverse data region in sequence, which can be considered an "implicit" AdaBoost. Moreover, considering that the two networks are always trained based on loss functions with different data weights, this interaction encourages the diversity of base networks.

To implement this updating rule, a simple method is to use function $w_1 = p_2/(p_1+p_2)$, and then assign $W_{1d} = w_1$ and $W_{2d} = 1-w_1$. Here, for convenience, we use p_1 and p_2 to represent the probabilities that the point $\boldsymbol{x}_d$ is classified by the two base networks correctly. However, this is problematic, as illustrated

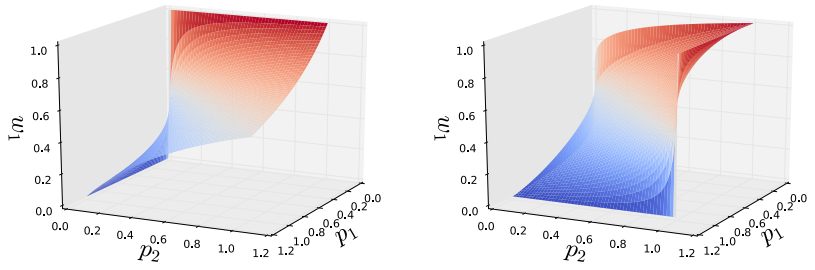

Figure 3: Function $w_1 = p_2/(p_1+p_2)$ (left) and function $w_1 = \ln(p_1)/(\ln(p_1)+\ln(p_2))$ (right), where $0 < p_1, p_2 < 1$.

on the upper panel of Fig. 3. For example, when both p_1 and p_2 are large and close to each other, w_1 will be close to 0.5. In this situation, there will be no big difference between W_{1d} and W_{2d}. In addition, this situation will occur frequently as neural networks with high flexibility will fit the data well. As a result, the function has difficulty to make a data point have different weights in two base networks.

Instead, we use function $w_1 = \ln(p_1)/(\ln(p_1) + \ln(p_2))$, as shown on the lower panel of Fig. 3, to update data weights. It is observed that the function is more sensitive to small differences between p_1 and p_2 when they are both large. Specifically, for $\{\boldsymbol{x}_d, \boldsymbol{y}_d\}$, we update its weights as

$$W_{1d}^{(n)} = \frac{\ln P(\boldsymbol{y}_d|\boldsymbol{x}_d, \boldsymbol{\theta}_1^{(n-1)})}{\ln P(\boldsymbol{y}_d|\boldsymbol{x}_d, \boldsymbol{\theta}_1^{(n-1)}) + \ln P(\boldsymbol{y}_d|\boldsymbol{x}_d, \boldsymbol{\theta}_2^{(n-1)})}, \tag{2}$$

$$W_{2d}^{(n)} = 1 - W_{1d}^{(n)}. \tag{3}$$

The optimization objective. After updating network parameters and data weights, we compute the optimization objective of two base networks. For each data point, we compute the mean value of output probabilities of the two networks as the final prediction, and the optimization objective (loss) in the nth

iteration is shown as follows:

$$L^{(n)} = -\sum_{d=1}^{D} \log \frac{P(\boldsymbol{y}_d|\boldsymbol{x}_d, \boldsymbol{\theta}_1^{(n)}) + P(\boldsymbol{y}_d|\boldsymbol{x}_d, \boldsymbol{\theta}_2^{(n)})}{2}, \tag{4}$$

where $L^{(n)}$ is a variable that we monitor at each iteration, and the two base networks corresponding to the minimal value of $L^{(n)}$ in the training process are what we aim to learn. It means that the optimization goal of InterBoost is to learn a combination of two base networks, in which either both base networks fit the training data well independently or one of them does not do it well but their combination does it well.

The training procedure of InterBoost is summarized in Algorithm 1. First, two base networks are trained by minimizing loss functions L_1 and L_2, respectively. Secondly, weights of the training data points are updated with Equations (2) and (3). The two steps are repeated until the maximum iteration number is reached. Finally, the two base networks for the minimal loss of InterBoost are obtained.

3.3. InterBoost prediction

Through the interactive and iterative training process, the two networks become well trained over various regions of the problem space represented by the data. In other words, they become "experts" with different knowledge. Therefore, we adopt a simple fusion strategy of linearly combining the prediction results of two networks with equal weights as the final prediction of InterBoost:

$$P\left(\boldsymbol{y} \mid \tilde{\boldsymbol{x}}\right) = \frac{P\left(\boldsymbol{y} \mid \tilde{\boldsymbol{x}}, \boldsymbol{\theta}_1'\right) + P\left(\boldsymbol{y} \mid \tilde{\boldsymbol{x}}, \boldsymbol{\theta}_2'\right)}{2}, \tag{5}$$

where $P\left(\boldsymbol{y} \mid \tilde{\boldsymbol{x}}, \boldsymbol{\theta}_i'\right)$, $i \in \{1, 2\}$ is the probability that unseen data point $\tilde{\boldsymbol{x}}$ is classified into class $\boldsymbol{y}$ by the ith base network. That is, prediction of InterBoost for unseen data synthesizes the the prediction results of two base networks.

3.4. Discussion of InterBoost

During the training process, we always keep the constraints $W_{1d} + W_{2d} = 1$ and $0 < W_{1d}, W_{2d} < 1$. Equations (2) and (3) are designed for updating

Algorithm 1 InterBoost training procedure

Input:

Training set $\{(\boldsymbol{x}_d, \boldsymbol{y}_d) \mid d \in \{1, \ldots, D\}\}$ and the maximum number of iterations N.

Steps:

Initialize weights for each data point, $W_{1d}^{(1)}$, $W_{2d}^{(1)}$, and parameters of two base networks $\boldsymbol{\theta}_1^{(0)}$, $\boldsymbol{\theta}_2^{(0)}$; n $\leftarrow$ 0; $L^{(0)} = \infty$.

repeat

n $\leftarrow$ n + 1

Update $\boldsymbol{\theta}_1^{(n)}$ and $\boldsymbol{\theta}_2^{(n)}$ by minimizing (1)

Update $W_{1d}^{(n+1)}$, $W_{2d}^{(n+1)}$, $d \in \{1, \ldots, D\}$, according to (2) and (3)

Computing the optimization objective, $L^{(n)}$, by (4)

if $L^{(n)} < L^{(n-1)}$ **then**

$\boldsymbol{\theta}_1' = \boldsymbol{\theta}_1^{(n)}$ and $\boldsymbol{\theta}_2' = \boldsymbol{\theta}_2^{(n)}$

end if

until n == N

return Parameters of two base networks, $\boldsymbol{\theta}_1'$ and $\boldsymbol{\theta}_2'$

weights of data points, so that the weight updating rule is sensitive to small differences between prediction probabilities from two base networks to encourage the diversity between base networks. Furthermore, if the prediction of a data point in one network is less accurate than another network, its weight in the next round will be larger than its weight for another network, thus making base networks focus on different regions continually.

Similar to Bagging and AdaBoost, our InterBoost has no limitation on the type of neural networks. In addition, it is straightforward to extend InterBoost to multiple networks, just by keeping $\sum_{i=1}^{H} W_{id} = 1$, $d \in \{1, \ldots, D\}$, in which H is the number of base networks and $0 < W_{id} < 1$. The reason why we chose two base networks in this work is purely for simplicity and less test cost.

4. Experimental results and discussion

We chose four small-sample datasets for image classification to evaluate the effectiveness of the proposed InterBoost, through: 1) comparing InterBoost with classical ensemble methods and new ensembles for networks on the four datasets; 2) evaluating their performance under different training sample sizes; 3) conducting the Wilcoxon signed-rank tests on experimental results to further show that the superiority of InterBoost is not due to chance; 4) analyzing the mechanism of InterBoost in terms of accuracy and diversity; and 5) discussing experimental results.

4.1. Datasets and data preprocessing

- LabelMe (LM): A subset of the scene classification dataset from [34]. The dataset contains 8 classes of natural scene images: coast, forest, highway, inside city, mountain, open country, street and tall building. We randomly selected 200 images for each class, so the total number of images is 1,600.
- UIUC-Sports (UIUC): An 8 class sports events classification dataset[1] from [33]. The dataset contains 8 classes of sports scene images. The total

[1] http://vision.stanford.edu/lijiali/Resources.html

Table 1: Comparison of classification performance on the UIUC, LM, 15Scenes (15Sce.) and Caltech101 (Calte.) datasets. Methods include Baseline (Base.), Dropout (Drop.), Augmentation (Aug.), Bagging (Bag.), AdaBoost (Ada.), MoE, Snapshot (Snap.) and the proposed InterBoost (Inter.). Each method runs 60 rounds, and the mean values and standard deviations (Std.) of classification accuracies are reported.

		Base.	**Drop.**	**Aug.**	**Bag.**	**Ada.**	**MoE**	**Snap.**	**Inter.**
UIUC	Mean	0.880	0.870	0.894	0.852	0.878	0.876	0.901	**0.904**
	Std.	0.025	0.025	0.025	0.035	0.033	0.016	0.005	**0.003**
LM	Mean	0.864	0.855	0.854	0.856	0.881	0.868	0.883	**0.890**
	Std.	0.037	0.048	0.039	0.033	0.014	0.019	0.008	**0.005**
15Sce.	Mean	0.833	0.822	-	0.821	0.835	0.832	0.843	**0.849**
	Std.	0.024	0.034	-	0.017	0.014	0.011	0.004	**0.003**
Calte.	Mean	0.878	0.881	-	0.873	0.906	0.891	0.934	**0.935**
	Std.	0.058	0.041	-	0.036	0.025	0.005	0.001	**0.001**

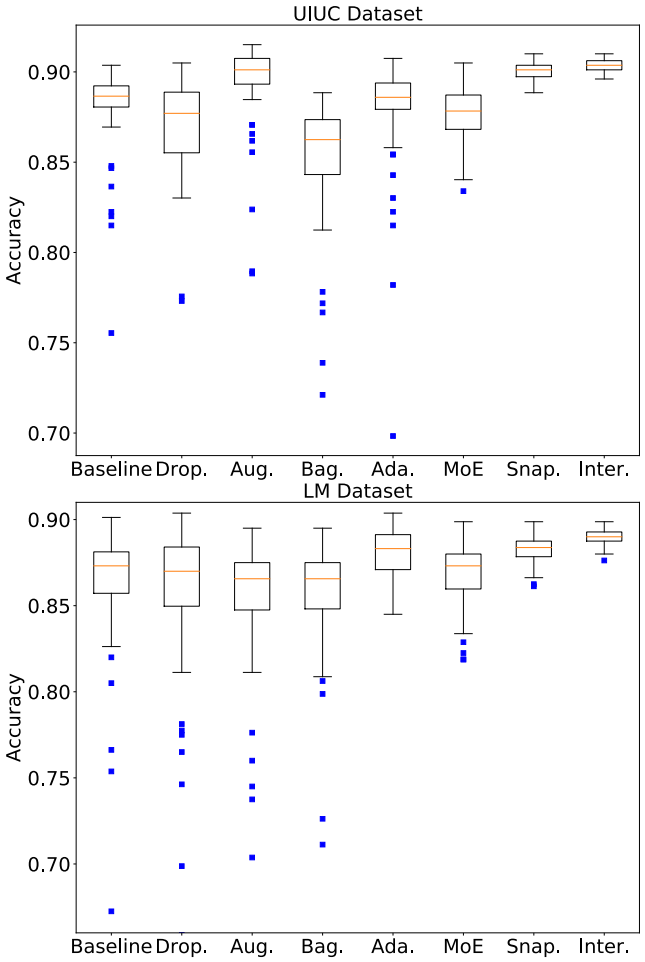

Figure 4: Comparison of the accuracies obtained by Baseline, Dropout (Drop.), Augmentation (Aug.), Bagging (Bag.), AdaBoost (Ada.), MoE, Snapshot (Snap.), and the proposed InterBoost method (Inter.) on the UIUC and LM datasets. Each method has been evaluated for 60 rounds, and the distributions of accuracies are shown via boxplots. In each boxplot, the central mark is the median; the edges of the box are the 25th and 75th percentiles, respectively; and the outliers are marked individually.

number of images is 1,579. The numbers of images for different classes are: bocce (137), polo (182), rowing (250), sailing (190), snowboarding (190), rock climbing (194), croquet (236) and badminton (200).

- 15Scenes [35]: A dataset of 15 classes of natural scene images: coast, forest, highway, inside city, mountain, open country, street and tall building. We randomly select 200 images for each class, so the total number of images is 3,000.

- Caltech101 [36]: A dataset of pictures of objects in 101 categories, and the size of each category is approximately 40-800 images. The total number of images is 4,000.

For the LM and 15Scenes datasets, each class in both training and test datasets contains 100 data points. For the UIUC and Caltech101 datasets, different classes have different sizes, and therefore we randomly and equally split the data in each class into training and test datasets. We resize the images on these four datasets into the same size of 256×256.

4.2. Implementation details of the compared methods

In order to evaluate the classification performance of the proposed InterBoost method, we compare InterBoost with 1) one convolutional neural network (CNN) with VGG16 styled (The method is the base of other compared methods, we call it Baseline); 2) Baseline with Dropout (Dropout); 3) Baseline with augmentation (Augmentation); 4) Bagging of Baseline (Bagging); 5) AdaBoost of Baseline (AdaBoost); 6) Mixture of experts (MoE); and 7) Snapshot Ensembling of Baseline (Snapshot) [32], on the four small-sample datasets.

Since Baseline is the basis of constructing other compared methods in our experiments, we first introduce the implementation of Baseline. Baseline adopted the VGG16 [5] style CNN, containing 13 convolutional layers and 2 fully connected layers with 1 hidden layer containing 32 hidden units. In the part of convolutional layers, the structure is the same as VGG16, and in the part of

Table 2: Comparison of classification accuracies obtained by Baseline (Base.), Dropout (Drop.), Augmentation (Aug.), Bagging (Bag.), AdaBoost (Ada.), MoE, Snapshot (Snap.) and the proposed InterBoost method (Inter.) on UIUC and LM datasets when the training datasets are reduced. Mean values and standard deviations (Std.) are listed in cells of the table. Each method runs 60 rounds on each dataset. The notation DatasetName-n denotes the configuration in which the training dataset in the named dataset is reduced by n data points for every class from the original training dataset, while the test datasets are unchanged.

		Base.	**Drop.**	**Aug.**	**Bag.**	**Ada.**	**MoE**	**Snap.**	**Inter.**
UIUC-20	Mean	0.855	0.856	0.886	0.836	0.862	0.863	0.882	**0.892**
	Std.	0.054	0.052	0.018	0.031	0.042	0.016	0.006	**0.003**
UIUC-30	Mean	0.819	0.826	0.866	0.776	0.814	0.850	0.866	**0.872**
	Std.	0.058	0.080	0.030	0.103	0.076	0.014	0.006	**0.005**
UIUC-40	Mean	0.806	0.824	0.845	0.794	0.823	0.827	0.846	**0.853**
	Std.	0.072	0.059	0.023	0.048	0.057	0.019	0.006	**0.005**
UIUC-50	Mean	0.787	0.806	0.830	0.730	0.734	0.807	0.830	**0.835**
	Std.	0.075	0.060	0.023	0.093	0.158	0.033	0.008	**0.006**
		Base.	**Drop.**	**Aug.**	**Bag.**	**Ada.**	**MoE**	**Snap.**	**Inter.**
LM-10	Mean	0.832	0.843	0.853	0.830	0.865	0.857	0.870	**0.880**
	Std.	0.068	0.061	0.043	0.516	0.023	0.019	0.007	**0.004**
LM-30	Mean	0.830	0.810	0.838	0.810	0.842	0.841	0.843	**0.854**
	Std.	0.028	0.087	0.024	0.042	0.018	0.019	0.009	**0.007**
LM-50	Mean	0.781	0.779	0.792	0.754	0.794	0.791	0.802	**0.810**
	Std.	0.033	0.063	0.027	0.074	0.021	0.018	0.009	**0.007**
LM-70	Mean	0.776	0.788	0.783	0.755	0.798	0.796	0.803	**0.819**
	Std.	0.042	0.061	0.035	0.045	0.026	0.025	0.016	**0.013**

fully connected layers, the activation function of the hidden layer was the Rectified Linear Unit function ($ReLU$), and the activation function of the output layer was *Softmax*. We used mini-batch gradient descent to minimize the softmax cross entropy loss. The optimization algorithm was RMSprop, the initial learning rate was 0.001, the coefficient of L_2 norm penalty on network weights is 0.001, the batch size was 32, and the epoch number was 100. In the training process, all the parameters in convolutional layers were directly initialized by those in the pre-trained VGG16 on the ImageNet dataset, and are then frozen during training and test processes.

Regarding Dropout and Augmentation, Baseline is the backbone network of them. For Dropout, we added a Dropout layer after the hidden layer in the fully connected parts of Baseline with a probability of 0.01. For Augmentation [54, 3], flip and shear transformation is adopted. In particular, we randomly flip images horizontally, and the shear intensity is 0.2.

Regarding Bagging, AdaBoost, MoE and Snapshot, Baseline is the base network (leaner) of them, and the number of base networks in all these methods is set to 2. For Bagging, instead of voting strategy, we added the predicting probabilities of the two base networks and then divided the sum by two as the final predicting probabilities. About MoE, we used two same Baseline networks as two base networks in MoE, and one gate network is implemented by sigmoid function. In Bagging, AdaBoost and MoE, the setting of parameters and optimization of base networks are set as the same with Baseline. For Snapshot, we followed the method of obtaining a snapshot network and changing the learning rate in [32], there are two iterations, and each iteration contains 50 epochs. Except for this, the setting of parameters and optimization of base networks in Snapshot is the same with Baseline.

Finally, for InterBoost, two base networks adopted the same network structure with Baseline. In addition, the iteration number of InterBoost was set as 5 and each iteration had 20 epochs on the UIUC and LM datasets; and the iteration number was set to 3 and each iteration had 33 epochs on the 15Scenes and Caltech101 datasets. The settings of iteration number and epoch number were

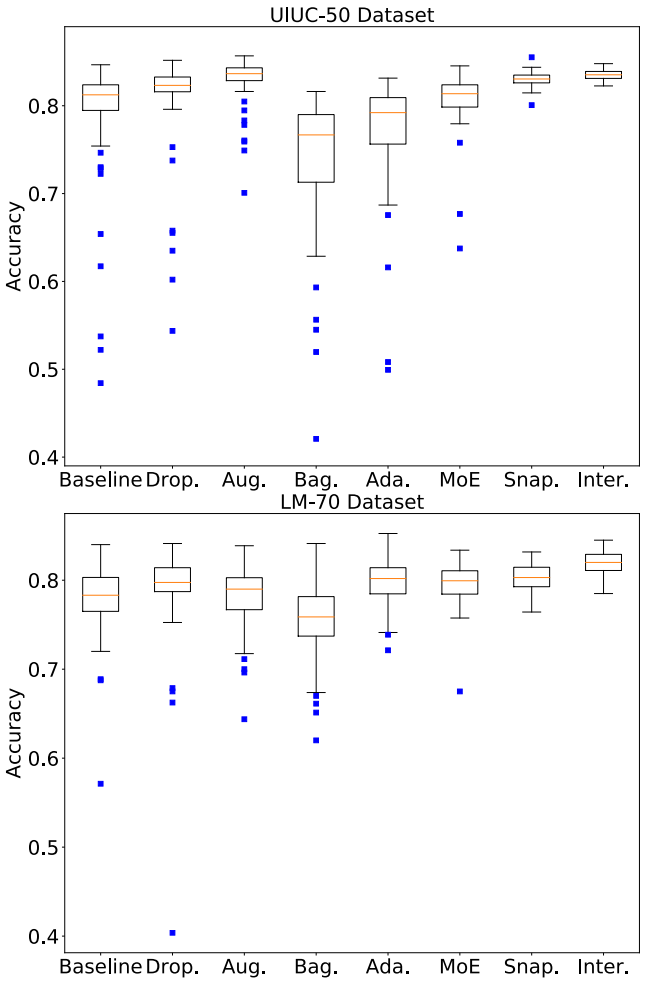

Figure 5: Comparison of the accuracies obtained by Baseline, Dropout (Drop.), Augmentation (Aug.), Bagging (Bag.), AdaBoost (Ada.), MoE, Snapshot (Snap.), and the proposed InterBoost method (Inter.) via boxplot on the UIUC-50 and LM-70 datasets. The rest of the caption is as in Fig. 4.

for two reasons: first, the sum of training epoch numbers of one base network in all iteration needs to be the same with Baseline; secondly, any base network can fit training data in each iteration. In addition, we adopted cyclic cosine annealing technique [32] for decreasing learning rate at each iteration.

All the methods were implemented on Keras [55].

4.3. Classification accuracies

We ran Baseline, Dropout, Augmentation, Bagging, AdaBoost, MoE, Snapshot Ensembling and InterBoost on the four datasets 60 rounds each. The mean value and standard deviation of accuracies are reported in Table 1: a larger mean value and a smaller standard deviation are better.

From Table 1, we can see that Dropout shows a slight improvement upon the performance of Baseline on the four datasets, while variances are larger. Augmentation does improve the the mean value due to increased data amount, however it does not reduce the variance. Bagging performs worse than the single base network. AdaBoost and MoE have very similar mean values to Baseline, but smaller variance than Baseline. Snapshot performs better than Baseline, Dropout and Augmentation, also better than Bagging, MoE and AdaBoost.

We can also see that our InterBoost has the best performance with an average accuracy of 90.36% on UIUC, 89.00% on LM, 84.88% on 15Scenes, and 93.45% on Caltech101. The average accuracy of our method is 2.4% absolutely higher than that of Baseline on UIUC, 3.6% absolutely higher than that of Baseline on LM, 1.5% absolutely higher than that of Baseline on 15Scenes, and 5.6% absolutely higher than that of Baseline on 15Scenes. In addition, our method has the smallest standard deviations 0.003, 0.005, 0.003, and 0.001 on UIUC, LM, 15Scenes and Caltech101, respectively, significantly outperforming all other compared methods.

To further evaluate the robustness and stability of the proposed method, in Fig. 4, we show boxplots of accuracies obtained by all the compared methods on the UIUC and LM datasets. We can see that among these seven methods, the boxes of Baseline are the largest on both datasets. The boxes of our Inter-

Boost are most compact, in which the maximum value and the lower quartile of accuracies are both higher than those of other compared methods. It is worth mentioning that there are some low-accuracy outliers in all other compared methods, but there is no low-accuracy outlier in our method.

In summary, our InterBoost demonstrates superior performance on LM, UIUC, 15Scenes and Caltech101; it also has smaller fluctuation than other methods, showing its ability to reduce variance in prediction for small-sample classification.

4.4. Classification accuracies for different training dataset sizes

It is well-known that when the number of training data points is reduced, the overfitting problem will get more severe. To further demonstrate the performance of our method in mitigating the overfitting problem, we reduced the size of the training datasets of UIUC and LM while keeping the sizes of the test datasets. For all experiments here, each method runs 60 rounds. For all methods, we do not change their settings, just run them on different training datasets, and then compute the mean values and standard deviations of accuracies as reported in Table 2, where the notation DatasetName-n denotes the configuration that the number of training data in the named dataset is reduced by n data points for every class from the original training dataset, while the test data are unchanged.

It can be observed from Table 2, with the decrease in size of training datasets, mean values of all the methods become smaller and variance of them become larger, however, the tendency among these methods is almost retained.

Compared with Baseline, Dropout has larger mean values in most of experiments, but only has smaller standard deviation on several sets of experiments; Augmentation has larger mean values and smaller standard deviations on UIUC and LM (see Fig. 5). It shows on these two datasets that Dropout has no big improvement for Baseline, and Augmentation could improve performance of Baseline when the training data size is reduced.

It is worth noting that three classical ensemble methods, Bagging, AdaBoost

Table 3: The p-values of our method versus other compared methods, Baseline (Base.), Dropout (Drop.), Augmentation (Aug.), Bagging (Bag.), AdaBoost (Ada.), MoE, Snapshot (Snap.), from the Wilcoxon signed-rank tests. Each method runs 60 rounds on each dataset. The notation DatasetName-n denotes the configuration in which the training dataset in the named dataset is reduced by n data points for every class from the original training dataset, while the test datasets are unchanged.

	Base.	**Drop.**	**Aug.**	**Bag.**	**Ada.**	**MoE**	**Snap.**
UIUC	7.21E-10	5.90E-14	4.55E-03	1.11E-18	7.04E-08	1.41E-18	2.55E-04
UIUC-20	4.90E-11	2.44E-11	2.04E-03	1.62E-11	2.50E-11	1.88E-11	4.17E-11
UIUC-30	2.44E-11	1.09E-09	**0.9120**	1.63E-11	2.69E-11	5.70E-11	1.88E-05
UIUC-40	3.82E-11	4.11E-10	**0.0690**	1.63E-11	1.23E-09	1.70E-11	7.13E-09
UIUC-50	2.01E-10	3.37E-07	**0.7183**	1.63E-11	1.80E-11	3.99E-11	6.00E-05
	Base.	**Drop.**	**Aug.**	**Bag.**	**Ada.**	**MoE**	**Snap.**
LM	1.18E-09	1.30E-09	4.20E-11	4.10E-11	2.28E-04	5.04E-10	1.49E-06
LM-10	6.41E-11	3.50E-08	8.08E-07	3.59E-11	7.44E-08	1.36E-10	6.27E-09
LM-30	3.51E-08	1.19E-06	1.19E-06	1.28E-06	2.28E-06	8.62E-06	1.21E-08
LM-50	1.10E-08	1.94E-06	7.56E-09	5.82E-11	1.18E-04	6.15E-09	2.55E-07
LM-70	1.28E-09	2.47E-06	1.97E-09	7.08E-11	2.53E-06	2.02E-07	9.81E-08
	Base.	**Drop.**	**Aug.**	**Bag.**	**Ada.**	**MoE**	**Snap.**
15Sce.	1.80E-11	2.21E-11	-	1.62E-11	3.91E-11	2.51E-11	4.30E-10
Calte.	1.63E-11	1.63E-11	-	1.63E-11	1.63E-11	1.62E-11	**0.1010**

and MoE, do not give large mean values and small variances. Snapshot, an ensemble method which is tailored for neural networks and constructs multiple base networks by finding different local minima of a loss function, performs better than the three classical ones. It also performs better than other Dropout and Augmentation. Finally, our method performs better than Snapshot.

In summary, compared to other methods, our InterBoost has the highest mean values and the lowest variances of accuracies on UIUC and LM with different reduced training dataset sizes.

4.5. Wilcoxon signed-rank tests

In the experiments above, our method shows superior performance to other methods on the sample mean and the sample standard deviation of classification accuracies. To further demonstrate these results are not due to chance, we did Wilcoxon signed-rank tests [37] for our method and other methods. A Wilcoxon signed-rank test is a non-parametric statistical hypothesis test that can be used to determine whether the mean difference between two sets of observations is zero. Therefore, the null-hypothesis of a Wilcoxon signed-rank test is that our method and the other method has the same mean value, and the corresponding p-values are listed in Table 3. (The p-values bigger than 0.01 are in bold.)

In Table 3, most of p-values are smaller than 0.01. It can be observed that on all datasets, the compared Baseline, Dropout, Bagging, AdaBoost, and MoE have extremely small p-values with our method under Wilcoxon signed-rank tests. It means that the null-hypothesis that our method and these six compared methods have the same mean are rejected.

Augmentation has a larger p-value than 0.01 with our method on UIUC-30, UIUC-40 and UIUC-50. Except for this, the p-values of Augmentation are smaller than 0.01. It means that the null-hypothesis, that our method and Augmentation have the same mean value, is not always rejected. A similar pattern happens with Snapshot but only on Caltech101.

All these results indicate that the classification performance obtained by the proposed InterBoost is statistically significantly better than the compared

methods, in general.

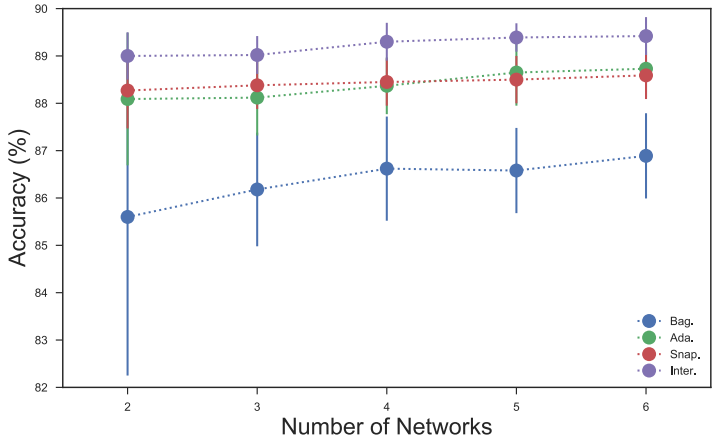

Figure 6: Comparison of the averaged accuracies obtained by Bagging (Bag.), AdaBoost (Ada.), Snapshot (Snap.), and the proposed InterBoost method (Inter.) on the LM dataset. Each method has been evaluated for 60 rounds.

4.6. The effect of the number of base networks

To further explore the the performance of InterBoost on multiple base networks, we increased the number of base networks to be 3, 4, 5 and 6, respectively, and compared the proposed InterBoost with Bagging, AdaBoost and Snapshot on the LM dataset. The weights of data points are updated by Equation 6.

$$W_{id}^{(n)} = \frac{\ln P(\boldsymbol{y}_d|\boldsymbol{x}_d, \boldsymbol{\theta}_i^{(n-1)})}{\sum_{i=1}^{K} \ln P(\boldsymbol{y}_d|\boldsymbol{x}_d, \boldsymbol{\theta}_i^{(n-1)})}, i \in \{1, 2, ..., K\}, \tag{6}$$

where K denotes the number of base networks. The classification accuracies are shown in Fig. 6.

From Fig. 6, we can observe the followings. Firstly, when the number of base networks increases, the InterBoost still performs best among all the methods. Secondly, with the increase of the number of base networks, the classification accuracies of all compared methods have been improved. Thirdly, such improvement, however, tends to be smaller as the number of the base network becomes

larger. A reason for this is, when the number of base networks become large, the diversity among base networks will become small.

4.7. Analysis on accuracy and diversity

From the experiments above, the proposed InterBoost outperformed Snapshot on the four small-sample image datasets. In this section, we aim to explain the reason for this improvement based on some quantitative analysis on the diversity (Kullback-Leibler divergence) and the accuracy (cross entropy loss and accuracy), and on the tendency of accuracy and diversity of two base networks in the training process of InterBoost. Following the definitions and notation in section 3, we compute the diversity (ambiguity) of two base networks, $D(\text{Base1}, \text{Base2})$, based on the Kullback-Leibler divergence:

$$D(\text{Base1}, \text{Base2}) = \sum_{d=1}^{D} KL_s[P(\boldsymbol{x}_d|\boldsymbol{\theta}_1), P(\boldsymbol{x}_d|\boldsymbol{\theta}_2)], \tag{7}$$

where $KL_s(P_1, P_2) = \frac{KL(P_1,P_2)+KL(P_2,P_1)}{2}$, and P_2 and P_1 represent two discrete probability distributions of the same dimension.

Comparison to Snapshot: To gain more insights of the difference between Snapshot and InterBoost, we compute the accuracies and cross entropy losses of Snapshot, InterBoost and their base networks, as well as the diversities (Equation 7) of the two base networks in Snapshot and InterBoost, respectively, on the training dataset of LM. The results are shown in Fig. 7. Since the accuracies of all of Snapshot, InterBoost and their base networks are 1.00, which means they correctly classify the training data, we did not show the accuracies.

From Fig. 7, it can be seen that: firstly, in Snapshot, the second base network has a lower boxplot of the cross entropy loss than the first network, which is mainly because the parameter initialization of the second base network uses the parameter values obtained from training the first base network. In contrast, the two base networks in InterBoost have similar box plots. In Snapshot and InterBoost, however, the boxplots of the cross entropy losses of all base networks are similar in general. Furthermore, the accuracies of all base networks in Snapshot and InterBoost are 1.00. Therefore, the accuracies of all base networks

in Snapshot and InterBoost are close to each other on the LM dataset. Secondly, from the lower panel of Fig. 7, on the training data of LM, the boxplot of $D(\text{Base1}, \text{Base2})$ of InterBoost is much higher than the one of Snapshot, and the minimum of $D(\text{Base1}, \text{Base2})$ in InterBoost is even higher than the maximum of $D(\text{Base1}, \text{Base2})$ in Snapshot.

In addition, we also show the confusion matrices of Snapshot, InterBoost and their base networks on the test dataset of LM in Fig. 8. The results shown are selected randomly from one round among the 60 rounds that Snapshot and InterBoost run. Snapshot and its two base networks have accuracies 87.87%, 88.00% and 88.12%, respectively; and InterBoost and its two base networks have accuracies 89%, 88.62% and 88.25%, respectively.

It can be observed that InterBoost has larger improvement upon its base networks than Snapshot. The reason is that the diversity of base networks in Snapshot is not enough, and when the diversity between base learners of an ensemble is small, the performance of an ensemble is close to the average of base learners [28], which explains the results in Fig. 8. In short, InterBoost has similar accuracies to Snapshot, and much larger diversity than Snapshot.

Analysis of InterBoost: To further explain how InterBoost obtains both accurate and diverse base networks on small-sample data, we monitored sample weights of Base1 in the training process of InterBoost, see Fig. 9. For clarity, we only show weights of training samples of the "coast" class before the 1st, 3rd and 5th iterations. We also show the accuracy and $D(\text{Base1}, \text{Base2})$ of InterBoost and its base networks at each iteration on the LM test dataset in Fig. 10.

From Fig. 9, we can see that when $n = 1$ (the weights are from random initialization), weights of most of samples in Base1 are close to 1 or 0, and with more iterations, weights of all samples are close to 0.5. Because $W_{1d} + W_{2d} = 1$ and $0 < W_{1d} < 1$, a similar pattern happens with Base2 in InterBoost. In addition, from Fig. 10, we can find with more iterations, the accuracies of two base networks show upward tendency and the diversities shows downward tendency.

Therefore, with more iterations, base networks can still keep certain diversity

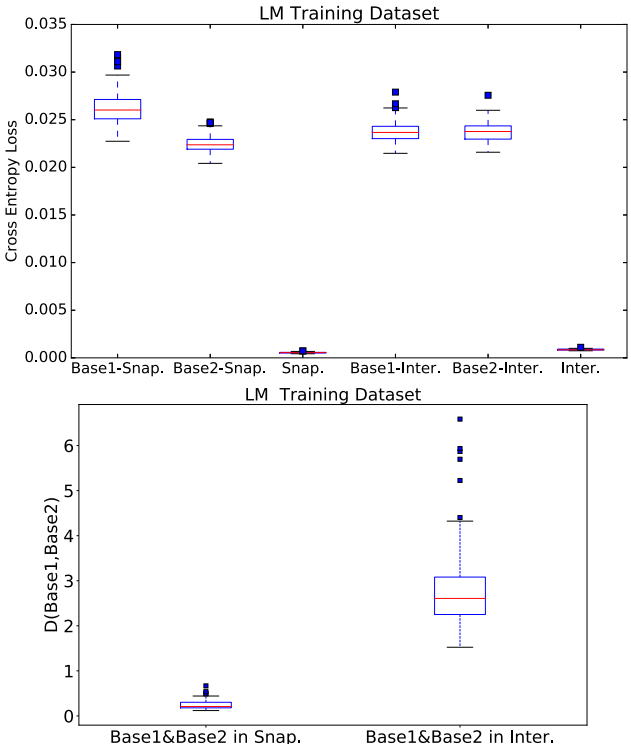

Figure 7: The cross entropy loss values of Snapshot, InterBoost and their base networks, and the diversity ($D(\text{Base1}, \text{Base2})$) between base networks on the training dataset of LM. In each boxplot, the central mark is the median, and the edges are the 25th and 75th percentiles. Each method runs 60 rounds.

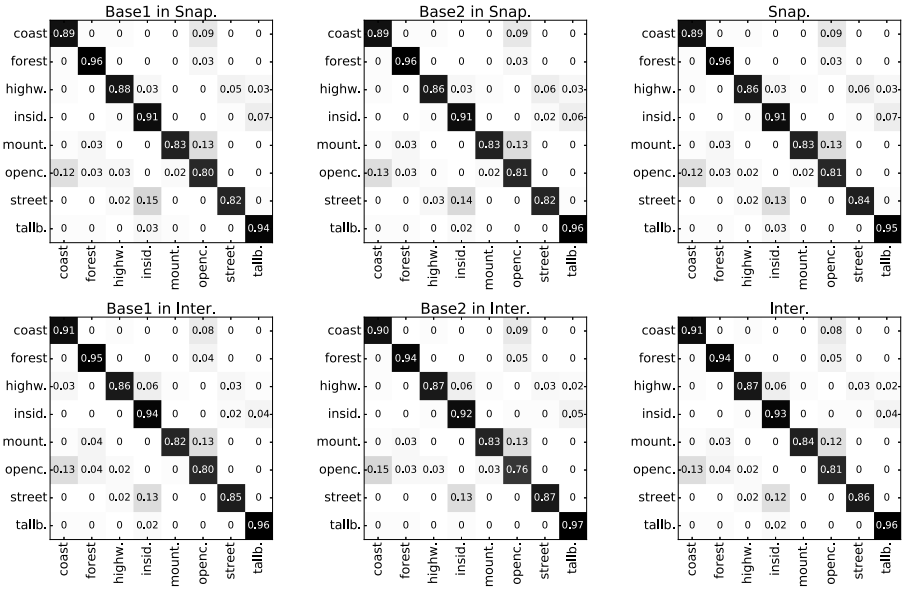

Figure 8: Confusion matrices of Snapshot (Snap.) and InterBoost (Inter.) and their base networks on the test data of LM. The accuracy of Snapshot is 87.87%, with 88.00% for its base network 1 (Base1) and 88.13% for its base network 2 (Base2). The accuracy of InterBoost is 89.00%, with 88.62% for its base network 1 and 88.25% for its base network 2.

(even though the diversity can become smaller gradually), while becoming more and more accurate, which illustrates that InterBoost can learn both accurate and diverse base networks and verifies the motivation for InterBoost.

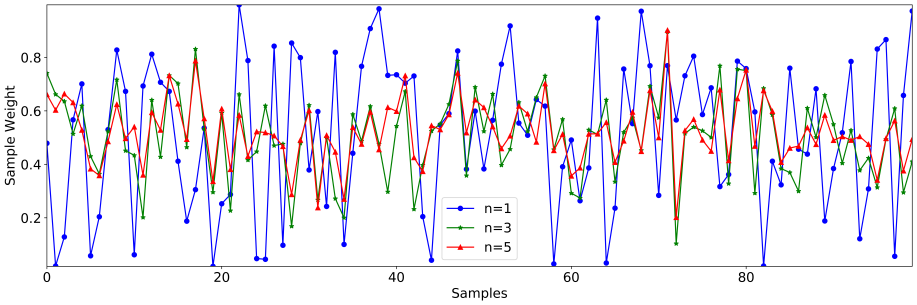

Figure 9: Sample weights of Base1, one base network in InterBoost, before each iteration of InterBoost, and n represents the iteration number. All the samples are from the "cost" class of training dataset, and the sample size of the "cost" class is 100. We randomly chose one round of the experiment of InterBoost on the LM dataset from the 60 rounds reported in Table 1.

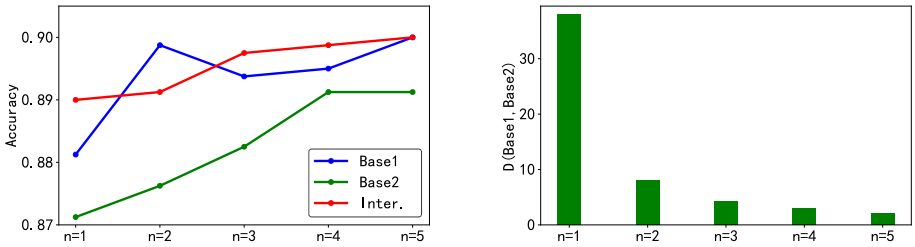

Figure 10: Accuracy of the base networks in InterBoost, and the diversity (D(Base1, Base2)) between two base networks in InterBoost on the test dataset of LM. Notation n represents the iteration number. The results are chosen from the same round as the one in Fig. 9.

4.8. Discussion

From the experimental results above, we find that InterBoost outperforms three classical ensemble methods (Bagging, AdaBoost, and MoE) and one newly proposed ensemble method, Snapshot, on small-sample image classification with

deep neural networks. As the bootstrap sampling applied in Bagging can generate two diverse training datasets, it can construct two diverse base networks. However, since bootstrap sampling samples data points with replacement, it makes the training data used for training base networks contain fewer distinct original data points, so that the base networks are not accurate enough in our small-sample experiments. As for AdaBoost, if the first classifier has no or few errors on the training data, little or no changes can be introduced to the weights of data points for training the second classifier. It will end up with training the base network twice on the original dataset and then combining their results. Therefore, in this situation, the diversity between the base networks cannot be ensured. Regarding MoE, its gating network is mainly responsible for assigning data to different subnetworks, and different subnetworks are used to fit different subsets on the training data. Since MoE is optimized based a single cross-entropy loss and does not pose any specific constraints on the subnetworks, it is difficult to ensure the accuracy of the subnetworks. Regarding Snapshot, the reason that InterBoost outperform it on small-sample image classification lies in the fact that InterBoost can not only make the base networks have similar accuracies to Snapshot, but also obtain larger diversity.

5. Conclusion

In the paper, we proposed an ensemble method called InterBoost to train neural networks for small-sample image classification. In the training procedure, the two base networks share information with each other, and are trained interactively and iteratively. The interaction between the base networks make each of them more accurate and make the diversity between them be kept as as large as possible. In the end, two diverse and accurate base network are obtained. Experimental results on four commonly used datasets demonstrated that InterBoost 1) can obtain two diverse base networks with good classification performance; 2) has better generalization performance than other ensemble methods; and 3) is statistically significantly better than the compared methods.

Future work on InterBoost includes increasing the number of its base networks, and extending it to different types of networks and different kinds of data.

6. Acknowledgements

This work was supported in part by the National Key R&D Program of China under Grant 2019YFF0303300 and under Subject II No. 2019YFF0303302, by the National Natural Science Foundation of China (NSFC) under Grant 61906080, Grant 61763028, Grant 61773071, Grant 61922015, and Grant U19B2036, by the National Science and Technology Major Program of the Ministry of Science and Technology under Grant 2018ZX03001031, by the Beijing Academy of Artificial Intelligence (BAAI) under Grant BAAI2020ZJ0204, by the Beijing Nova Programme Interdisciplinary Cooperation Project under Grant Z191100001119140, by the Key Program of Beijing Municipal Natural Science Foundation under Grant L172030, and by the Hong-liu Outstanding Youth Talents Foundation of Lanzhou University of Technology.

References

[1] Y. Luo, Y. Wen, D. Tao, J. Gui, C. Xu, Large margin multi-modal multi-task feature extraction for image classification, IEEE Transactions on Image Processing 25 (1) (2016) 414–427.

[2] G. Cheng, P. Zhou, J. Han, Duplex metric learning for image set classification, IEEE Transactions on Image Processing 27 (1) (2018) 281–292.

[3] B. Shi, X. Bai, C. Yao, An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition, IEEE transactions on pattern analysis and machine intelligence 39 (11) (2017) 2298–2304.

[4] A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

[5] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014).

[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.

[7] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436.

[8] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, T. Lillicrap, One-shot learning with memory-augmented neural networks. arxiv preprint, arXiv preprint arXiv:1605.06065 (2016).

[9] X. Dong, L. Zheng, F. Ma, Y. Yang, D. Meng, Few-example object detection with model communication, IEEE Transactions on Pattern Analysis and Machine Intelligence 41 (7) (2018) 1641–1654.

[10] J. Shu, Z. Xu, D. Meng, Small sample learning in big data era, arXiv preprint arXiv:1808.04572 (2018).

[11] S. Rahman, S. Khan, F. Porikli, A unified approach for conventional zero-shot, generalized zero-shot, and few-shot learning, IEEE Transactions on Image Processing 27 (11) (2018) 5652–5667.

[12] L. Fei-Fei, R. Fergus, P. Perona, One-shot learning of object categories, IEEE transactions on pattern analysis and machine intelligence 28 (4) (2006) 594–611.

[13] F. Zhu, Z. Ma, X. Li, G. Chen, J.-T. Chien, J.-H. Xue, J. Guo, Image-text dual neural network with decision strategy for small-sample image classification, Neurocomputing 328 (2019) 182–188.

[14] M. Long, H. Zhu, J. Wang, M. I. Jordan, Deep transfer learning with joint adaptation networks, in: International Conference on Machine Learning, 2017, pp. 2208–2217.

[15] C. Zhang, S. Bengio, M. Hardt, B. Recht, O. Vinyals, Understanding deep learning requires rethinking generalization, arXiv preprint arXiv:1611.03530 (2016).

[16] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, Journal of Machine Learning Research 15 (1) (2014) 1929–1958.

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

[18] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, L. Song, Sphereface: Deep hypersphere embedding for face recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, Vol. 1, 2017, p. 1.

[19] W. Wan, Y. Zhong, T. Li, J. Chen, Rethinking feature distribution for loss functions in image classification, in: IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 9117–9126.

[20] S. Kim, D. Min, S. Kim, K. Sohn, Feature augmentation for learning confidence measure in stereo matching, IEEE Transactions on Image Processing 26 (12) (2017) 6019–6033.

[21] Z. Zhong, L. Zheng, Z. Zheng, S. Li, Y. Yang, Camstyle: A novel data augmentation method for person re-identification, IEEE Transactions on Image Processing 28 (3) (2019) 1176–1190.

[22] J. Salamon, J. P. Bello, Deep convolutional neural networks and data augmentation for environmental sound classification, IEEE Signal Processing Letters 24 (3) (2017) 279–283.

[23] W. Ouyang, X. Wang, C. Zhang, X. Yang, Factors in finetuning deep model for object detection with long-tail distribution, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 864–873.

[24] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, in: Advances in neural information processing systems, 2014, pp. 3320–3328.

[25] M. Oquab, L. Bottou, I. Laptev, J. Sivic, Learning and transferring mid-level image representations using convolutional neural networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 1717–1724.

[26] A. Krogh, J. A. Hertz, A simple weight decay can improve generalization, in: Advances in neural information processing systems, 1992, pp. 950–957.

[27] X. Li, L. Yu, D. Chang, Z. Ma, J. Cao, Dual cross-entropy loss for small-sample fine-grained vehicle classification, IEEE Transactions on Vehicular Technology 68 (5) (2019) 4204–4212.

[28] A. Krogh, J. Vedelsby, Neural network ensembles, cross validation, and active learning, in: Advances in neural information processing systems, 1995, pp. 231–238.

[29] E. van den Berg, B. Ramabhadran, M. Picheny, Training variance and performance evaluation of neural networks in speech, in: Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on, IEEE, 2017, pp. 2287–2291.

[30] P. M. Granitto, P. F. Verdes, H. A. Ceccatto, Neural network ensembles: evaluation of aggregation algorithms, Artificial Intelligence 163 (2) (2005) 139–162.

[31] Z.-H. Zhou, Ensemble methods: foundations and algorithms, CRC press, 2012.

[32] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, K. Q. Weinberger, Snapshot ensembles: Train 1, get m for free, in: International Conference on Learning Representations (ICLR), 2017.

[33] L.-J. Li, L. Fei-Fei, What, where and who? classifying events by scene and object recognition, in: IEEE 11th International Conference on Computer Vision, 2007, pp. 1–8.

[34] B. C. Russell, A. Torralba, K. P. Murphy, W. T. Freeman, LabelMe: a database and web-based tool for image annotation, International Journal of Computer Vision 77 (1) (2008) 157–173.

[35] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: In Computer Vision and Pattern Recognition (CVPR), IEEE, 2006, pp. 2169–2178.

[36] L. Fei-Fei, R. Fergus, P. Perona, Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories, in: 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPRW 2004, IEEE Computer Society, 2004.

[37] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics bulletin 1 (6) (1945) 80–83.

[38] E. L. Lehmann, J. P. Romano, Testing statistical hypotheses, Springer Science & Business Media, 2006.

[39] Z.-H. Zhou, J. Feng, Deep forest: towards an alternative to deep neural networks, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, AAAI Press, 2017, pp. 3553–3559.

[40] H. Schwenk, Y. Bengio, Boosting neural networks, Neural Computation 12 (8) (2000) 1869–1887.

[41] M. Moghimi, S. J. Belongie, M. J. Saberian, J. Yang, N. Vasconcelos, L.-J. Li, Boosted convolutional neural networks., in: The British Machine Vision Conference, 2016.

[42] Q. Miao, Y. Cao, G. Xia, M. Gong, J. Liu, J. Song, RBoost: label noise-robust boosting algorithm based on a nonconvex loss function and the numerically stable base learners, IEEE transactions on neural networks and learning systems 27 (11) (2016) 2216–2228.

[43] I. Goodfellow, Y. Bengio, A. Courville, Y. Bengio, Deep learning, Vol. 1, MIT press Cambridge, 2016.

[44] S. Mandt, M. D. Hoffman, D. M. Blei, Stochastic gradient descent as approximate bayesian inference, The Journal of Machine Learning Research 18 (1) (2017) 4873–4907.

[45] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, G. E. Hinton, Adaptive mixtures of local experts, Neural computation 3 (1) (1991) 79–87.

[46] S. E. Yuksel, J. N. Wilson, P. D. Gader, Twenty years of mixture of experts, IEEE transactions on neural networks and learning systems 23 (8) (2012) 1177–1193.

[47] S. Masoudnia, R. Ebrahimpour, Mixture of experts: a literature survey, Artificial Intelligence Review 42 (2) (2014) 275–293.

[48] Z. Ma, Y. Lai, W. B. Kleijn, Y.-Z. Song, L. Wang, J. Guo, Variational Bayesian learning for Dirichlet process mixture of inverted dirichlet distributions in non-gaussian image feature modeling, IEEE Transactions on Neural Networks and Learning Systems 30 (2) (2018) 449–463.

[49] Z. Ma, A. E. Teschendorff, A. Leijon, Y. Qiao, H. Zhang, J. Guo, Variational bayesian matrix factorization for bounded support data, IEEE transactions on pattern analysis and machine intelligence 37 (4) (2015) 876–889.

[50] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, R. Fergus, Regularization of neural networks using dropconnect, in: Proceedings of International Conference on Machine Learning, 2013, pp. 1058–1066.

[51] G. Huang, Y. Sun, Z. Liu, D. Sedra, K. Q. Weinberger, Deep networks with stochastic depth, in: European Conference on Computer Vision, Springer, 2016, pp. 646–661.

[52] M. Alan, D. M. George, Boosted residual networks, in: The 18th International Conference of Engineering Applications of Neural Networks, EANN 2017, Springer, Cham, 2017, pp. 137–148.

[53] S. Laine, T. Aila, Temporal ensembling for semi-supervised learning, in: International Conference on Learning Representations (ICLR), 2017.

[54] Y.-C. Chen, X. Zhu, W.-S. Zheng, J.-H. Lai, Person re-identification by camera correlation aware feature augmentation, IEEE transactions on pattern analysis and machine intelligence 40 (2) (2018) 392–408.

[55] F. Chollet, Keras:deep learning library for Python. runs on TensorFlow, Theano, or CNTK, URL https://github.com/fchollet/keras (2015).