SS-HCNN: Semi-Supervised Hierarchical Convolutional Neural Network for Image Classification

Tao Chen, Shijian Lu, Jiayuan Fan

Abstract

The availability of large-scale annotated data and uneven separability of different data categories become two major impediments of deep learning for image classification. In this paper, we present a Semi-Supervised Hierarchical Convolutional Neural Network (SS-HCNN) to address these two challenges. A large-scale unsupervised maximum margin clustering technique is designed, which splits images into a number of hierarchical clusters iteratively to learn cluster-level CNNs at parent nodes and category-level CNNs at leaf nodes. The splitting uses the similarity of CNN features to group visually similar images into the same cluster, which relieves the uneven data separability constraint. With the hierarchical cluster-level CNNs capturing certain high-level image category information, the category-level CNNs can be trained with a small amount of labelled images, and this relieves the data annotation constraint. A novel cluster splitting criterion is also designed which automatically terminates the image clustering in the tree hierarchy. The proposed SS-HCNN has been evaluated on the CIFAR-100 and ImageNet classification datasets. Experiments show that the SS-HCNN trained using a portion of labelled training images can achieve comparable performance with other fully trained CNNs using all labelled images. Additionally, the SS-HCNN trained using all labelled images clearly outperforms other fully trained CNNs.

Index Terms

SS-HCNN, semi-supervised, Hierarchical, unsupervised, image classification

Tao Chen (E-mail: ntuchentao@gmail.com, tel:(65)96476618) is with the Huawei Research Center, Singapore. Jiayuan Fan (E-mail: fanj@i2r.a-star.edu.sg, tel:(65)64082404) is with the Satellite Department in the Institute for Infocomm Research. Shijian Lu (E-mail: Shijian.Lu@ntu.edu.sg, tel:(65)91283996) is with the School of Computer Science and Engineering, Nanyang Technological University.

I. Introduction

The deep convolutional neural network (CNN) has been developed in various image classification applications [1], [2], [3], [4], [5], [6], [7], [8]. On the other hand, the deployment of deep CNN is facing two critical challenges: annotation of large-scale image datasets and uneven image separability across different object categories, e.g., "dog" and "sheep" images share higher visual similarity and are more difficult to separate, whereas "person" and "car" images share lower visual similarity and are much easier to differentiate. The traditional flat N-way CNN [1], [2] does not consider such uneven separability and often leads to sub-optimal object classification performance.

We propose a Semi-Supervised Hierarchical Convolutional Neural Network (SS-HCNN) that aims to address the image annotation constraint and category-wise uneven separability challenge. The idea is to partition images into a hierarchy of clusters through unsupervised clustering of the low-level features, and accordingly train a hierarchy of CNNs at root and parent nodes by using the generated cluster labels. The clusters at leaf nodes are more compact and consist of visually similar images of a small number of object categories, where CNNs can be trained effectively by using a small amount of image annotations. The SS-HCNN therefore consists of two training stages. The first is unsupervised which iteratively partitions images into clusters through clustering of the low-level image features and trains CNNs at root and parent nodes by using the generated cluster labels as the ground truth. The clustering process is iterative and terminates automatically according to a defined criterion. It relieves the uneven image separability constraint by clustering visually similar images into the same cluster where dedicated CNN can be trained for better classification performance. The second is supervised which trains category-level CNNs at leaf nodes for discriminative image classification. As the CNNs at parent nodes perform certain high-level coarse classification of images, the leaf node clusters are more compact and consist of images of a much smaller number of categories as compared with the original image set. Therefore, the category-level CNNs can be trained using a small amount of labelled training images and this relieves the data annotation challenge greatly.

The contributions of this work are threefold. First, it proposes a two-stage semi-supervised CNN learning framework that addresses the uneven image separability and image annotation constraints simultaneously, and demonstrates its superior performance in different image classification tasks. Second, it proposes a large-scale unsupervised maximum margin clustering

technique that employs the minibatch strategy to cluster the fully connected (FC) image features for hierarchical cluster-level CNN learning. Third, it designs a novel cluster splitting criterion which terminates the hierarchical clustering process automatically based on the underlying visual and structural image similarity. A voting based image scoring function is designed which classifies images by combining output of an ensemble of multiple leaf node CNNs.

II. Related Works

A. Semi-Supervised CNN Learning

Due to the challenges in collecting large-scale datasets and manual data annotation, semi-supervised CNN learning that uses partially labelled data samples has attracted increasing interest in recent years [4], [9], [10], [11], [12], [13]. Sparse Laplacian filter learning is adopted to obtain the network filters with unlabelled data for vehicle type classification [4]. In [9], a semi-supervised regularizer is added in the hidden or output loss layer or another separate auxiliary network sharing the first several layers with the original CNN in the deep structure. In [10], [11], they combine text region embeddings of variable sizes in the form of Long Short-Term Memory (LSTM) and convolution layers trained on the unlabelled data for text categorization. In [12], an online Expectation-Maximization (EM) method is developed to train deep CNN models from weakly annotated data. The method alternates between estimating the latent pixel labels and optimizing the DCNN parameters using stochastic gradient descent (SGD). In [13], an interesting model is proposed which first uses random noise to supervise the CNN pre-training and then learns CNN features through stochastic gradient descent (SGD) in an unsupervised manner. A common constraint of these works is that they adopt the flat N-way CNN as the learning structure where the uneven data separability problem is not well addressed.

B. Hierarchical CNN Learning

Several hierarchical learning methods have also been developed in recent years to address the uneven data separability problem [14], [15], [16], [17], [18], [19]. In [14], the label relationships are encoded in a hierarchical tree to improve the object classification accuracy. In [15], a hierarchy of CNNs is introduced with only two coarse categories due to scalability issue. The work [16] builds a hierarchical CNN with the main objective of transferring knowledge from a large network to a small network to achieve scalability. In [17], a two-level hierarchical CNN is designed to separate easy classes using a coarse category classifier and difficult classes using fine category

classifiers. In [18], a multi-level deep decision neural network is built where each node in the tree is a CNN. In [19], a two-level tree-structured network architecture is designed, which contains a generalist network producing coarse grouping of classes, and a set of expert networks for recognition of classes within each group. Crucially, the partition of categories is learned simultaneously with the parameters of the network trunk and the experts are trained jointly by minimizing a single learning objective over all classes.

These hierarchical CNN methods suffer from three typical limitations. First, they identify the image category hierarchy by performing spectral clustering on category confusion matrix [14], [15], [16], [17], [18], which is fully supervised requiring annotations of all training images. Second, they perform category-level clustering by grouping several fine image categories into a single coarse image category [14], [15], [16], [17], [18], [19], which will introduce larger variations. In addition, the clustering from fine categories into coarse categories may lead to misclassification of test images once they are classified into an incorrect coarse category at the beginning. Third, they either manually specify the depth of the hierarchy tree [17] or terminate the clustering according to the validation performance [18], and produce a flat tree where all the leaf nodes have the same depth. This ignores the fact that different clusters have different diversity and should be split into leaf nodes of different granularities and depth.

The proposed SS-HCNN addresses the above-mentioned constraints from several aspects. First, it adopts a hierarchical structure and mitigates the uneven data separability problem effectively. Second, the SS-HCNN performs unsupervised image clustering based on the underlying image feature similarity without requiring image labels. With the learned cluster-level CNNs at parent nodes which perform certain high-level classification tasks, only a small amount of labelled images are needed to train the category-level CNNs at leaf nodes. Third, instead of clustering based on image labels as in [17], [18], the SS-HCNN clusters images based on the underlying image feature similarity where images of the same object category may be clustered into different coarse categories. As a result, the risk of classifying a test image into an incorrect coarse category at the early stage is reduced because different coarse categories (clusters) under SS-HCNN may contain images of the same object category. Finally, the SS-HCNN designs an automatic and adaptive cluster splitting mechanism to address the image uneven separability issue. According to the underlying image feature similarity, the SS-HCNN clustering will stop early for images with high separability, e.g. “people” and “car” images, but continues to deeper layers for images with lower separability, e.g. “dog” and “sheep” images.

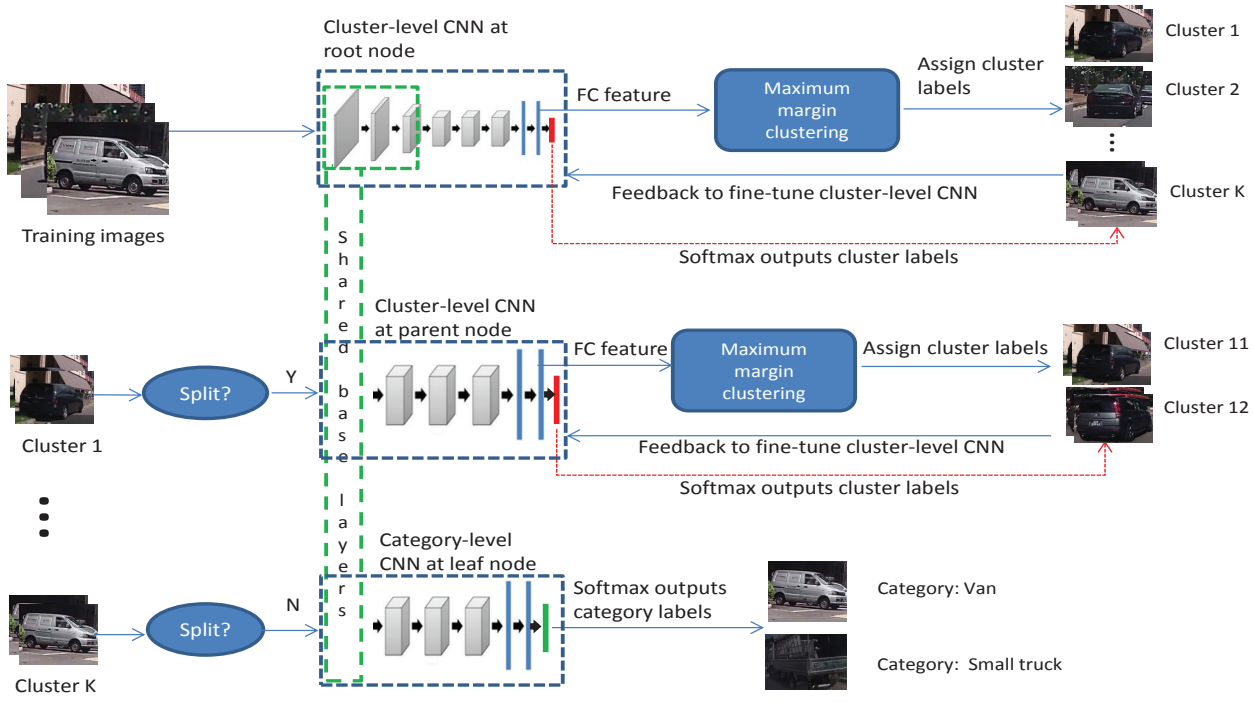

Fig. 1. Overview of the SS-HCNN layer structure and image clustering process at different nodes.

III. The Proposed Method

A. SS-HCNN Overview

The SS-HCNN is a tree structured deep hierarchical CNN, where each parent node corresponds to a cluster-level CNN which is trained through unsupervised clustering, and each leaf node corresponds to a category-level CNN which is trained through supervised learning. Fig. 1 shows an overview of the SS-HCNN layer structure and image clustering process at different nodes.

A pre-trained CNN model such as VGG [2] or ResNet [3] is first employed as the root node CNN to extract the fully connected (FC) features from each training image. The extracted FC features are then partitioned into multiple clusters through maximum margin clustering (MMC) [20] which assigns a cluster label to each training image. To avoid the time-consuming MMC clustering process while classifying a test image, a new softmax layer (red bar in Fig. 1) is added to train the root CNN to learn the correlation between each FC feature vector and the corresponding cluster label as generated by MMC. Therefore, the fine-tuning of the root node CNN (from the initial pre-trained CNN model) is unsupervised which uses the MMC-generated cluster labels as ground-truth labels. During the testing stage, the fine-tuned root node CNN can

thus predict a cluster label for each test image as indicated by the red dotted line in Fig. 1.

Note that using a pre-trained model is a widely adopted transfer learning practice since networks trained using different image datasets usually have similar lower layers which largely capture low-level features such as edges and corners [18], [21], [22]. In the SS-HCNN, we borrow the lower layers of a pre-trained CNN model to extract convolutional features for MMC. The MMC-produced cluster labels are then used as the ground truth to fine-tune the pre-trained CNN model so that it can produce consistent label predictions while classifying a new image during the testing phase.

A splitting criterion is designed to control the splitting of all clusters as generated by the root node, i.e. Cluster 1, Cluster 2, ..., Cluster K as shown in Fig. 1. If further splitting is needed, each generated cluster will go through the same MMC clustering and CNN fine-tuning process as the root node, which further produces child clusters, e.g., Cluster 11, Cluster 12 as illustrated in Fig. 1. Otherwise, the current cluster, e.g. Cluster K in Fig. 1, becomes a leaf cluster which contains images that have similar appearance, much smaller feature divergence and a much smaller number of image categories as compared with the original image dataset. A category-level CNN is finally trained by using a certain portion of leaf cluster images together with their annotations, where a new softmax layer (green bar in Fig. 1) is added for training. It is usually more accurate and reliable when a larger portion of the leaf cluster images together with their annotations are used for training. During the testing phase, the category-level CNN at a leaf node will predict a category label for each test image that is routed to that leaf node.

B. Large-scale Unsupervised Hierarchy Learning

The purpose of learning a CNN hierarchy is two folds. First, it targets to group visually similar images of different categories into the same coarse cluster and train dedicated cluster-level CNNs for better classification of the clustered images, hence addresses the uneven image separability effectively. Second, it trains a set of cluster-level CNNs that can perform high-level classification of test images. As a result, only a small amount of annotated images are needed to train the category-level CNNs at leaf nodes which relieves the image annotation challenge greatly.

We derive the image cluster labels by employing the MMC [20] which is an extension of the supervised large margin theory to the unsupervised scenario. The MMC optimizes the linear models learned for each cluster and simultaneously classifies each sample into a cluster, often leading to more compact clusters than other graph or spectral based methods [17], [23]. On the

other hand, the original MMC is more suitable for small-scale data clustering due to the large margin optimization. We propose a large-scale MMC technique by incorporating the minibatch idea that is widely used in CNN training [1] as described below.

Suppose $\{\mathbf{f}_i\}_{i=1}^{M}$ denote the FC feature vectors extracted from the first minibatch of training images, where M is the minibatch size which is experimentally set at 1024, according to the compromise between the clustering speed and clustering accuracy. The MMC first identifies K clusters from the M feature vectors by solving the following objective function,

$$\min_{W,Y,\xi\geq 0} \quad \left\{ \frac{1}{2}\sum_{k=1}^{K} \|\mathbf{w}_k\|^2 + \frac{C}{K}\sum_{i=1}^{M}\sum_{j=1}^{K}\xi_{ij} \right\} \tag{1}$$

$$\begin{aligned} s.t. \quad & \sum_{k=1}^{K} y_{ik}\mathbf{w}_k^T\mathbf{f}_i - \mathbf{w}_j^T\mathbf{f}_i \geq 1 - y_{ij} - \xi_{ij}, \quad \forall i,j \\ & y_{ik}\in\{0,1\}, \sum_{k=1}^{K} y_{ik} = 1 \qquad\qquad \forall i,k \\ & L \leq \sum_{i=1}^{M} y_{ik} \leq U, \qquad\qquad \forall k \end{aligned} \tag{2}$$

where $W = \{\mathbf{w}_k\}, k = 1, ..., K$ are the learned optimal linear models for the K clusters. The $Y = \{y_{ik}\}, i = 1, ..., M, k = 1, ..., K$ are the assigned cluster labels, where $y_{ik} = 1$ indicates that the i-th training sample is clustered into the k-th cluster. The $\xi = \{\xi_{ij}\}, i = 1, ..., M, j = 1, ..., K$ are slack variables to allow soft margin, and C is a trade-off parameter. The second constraint in Eq. 2 ensures that each training sample will be clustered into only one cluster. The last constraint controls the sample size of the k-th cluster to be between a lower bound L and an upper bound U, which generates a set of balanced clusters with moderate sample size. The parameters L and U are set at $0.9\frac{M}{K}$ and $1.1\frac{M}{K}$ respectively, by grid search that varies the multiplier coefficient between 0 and 2 with a step size of 0.1.

The second minibatch is then clustered into K clusters by MMC in the similar way. We merge each newly generated cluster with one of the K clusters as generated from the first minibatch based on the cluster similarity. In particular, the cluster similarity is computed by a matching score s_{kj} between each newly generated cluster j and previously generated cluster k as follows:

$$s_{kj} = \frac{1}{N_j}\sum_{i=1}^{N_j} \mathbf{w}_k^T\mathbf{f}_i \tag{3}$$

where N_j is sample number of the newly generated cluster j, and $\mathbf{f}_i$ is the i-th feature vector within it. It can be seen that the matching score is computed by the average likelihood score of the FC feature vector $\mathbf{f}_i$ in the newly generated cluster j as evaluated by the previously trained

cluster model $\mathbf{w}_k$. The newly generated cluster j is thus merged into the previously generated cluster k that produces the largest matching score s_{kj} as defined in Eq. 3.

The merged cluster acts as the new cluster k for the merging of the later clusters. To avoid the complex cluster model re-training, we keep both linear models $\mathbf{w}_k$ and $\mathbf{w}_j$ from clusters k and j and use an ensemble of the two component models for merging newly generated clusters. From the third image minibatch onwards, the matching score s_{kj} will thus be computed as follows:

$$s_{kj} = \frac{1}{N_j} \sum_{i=1}^{N_j} \left(\max_{p=1,\ldots,P} \left(\mathbf{w}_p^T f_i \right) \right) \tag{4}$$

where P is the number of the accumulated component models in the merged cluster k. It can be seen that the maximum matching score across the ensemble component models is used to decide the merging of the new clusters as generated from the third minimatch and thereafter. At the end, each new cluster as generated from the ensuing image minibatches is linked to one of the previous K clusters as defined in Eq. 4, and all images in the large-scale dataset will be assigned with a cluster label. These cluster labels will be used as the ground truth to fine tune the cluster-level CNNs, to be detailed in Section 3.4.

For clustering of FC features extracted from large-scale datasets, the proposed minibatch based MMC is advantageous due to its high computational efficiency and manageability. As a comparison, global clustering such as k-means clustering is much more challenging because it is memory inefficient and computationally expensive - imagine loading million-scale vectors with each vector of thousands dimension into the memory and doing k-means on them. Additionally, global clustering is more liable to produce unbalanced clusters with very large or small sizes. The minibatch based MMC clustering aims to capture the overall underlying image similarity which may not always cluster an image into the correct cluster. On the other hand, it will always route an image to one leaf node cluster where the image category information will be captured by the category-level CNN. Given a new test image, we also design an image scoring scheme which combines outputs of multiple leaf-node CNNs to address the possible image mis-clustering problem (more details to be presented in Section 3.5).

Note that unlike flat CNNs such as [13], updating clusters during the CNN fine-tuning does not introduce much change to the clusters in SS-HCNN. To verify this, we conducted a new test by updating clusters while fine-tuning the cluster-level CNN, and the study shows that the clusters have little change throughout the iterative CNN fine-tuning process. The little cluster change is largely due to two reasons: 1) the cluster-level CNN in SS-HCNN is pre-trained

using correct category labels, which has better cluster prediction capability than [13] that starts with random noise for supervision of the CNN pre-training; 2) SS-HCNN adopts a hierarchical structure where cluster-level CNN only predicts 3 or 4 (for CIFAR or ImageNet to be discussed later) cluster labels which is very coarse categorization and less prone to errors, whereas flat CNNs such as [13] predict a much larger number of fine categories.

C. Cluster Splitting

Determining when to stop cluster splitting is important to control the depth of the tree-structured CNN hierarchy. Over-splitting of a cluster brings little accuracy gain but increases computational costs greatly. We design a cluster splitting metric that considers both the cluster size and the feature divergence as defined by,

$$d = \mathbf{I}\left(N_k > t\right) \cdot \log \left(\frac{\sum_{i=1}^{N_k} \left(\max_{p=1,\ldots,P} \left(\mathbf{w}_p^T f_i \right) \right)}{\sum_{i=1}^{N_k} \left(\min_{p=1,\ldots,P} \left(\mathbf{w}_p^T f_i \right) \right)} \right) \tag{5}$$

where d is the metric score which controls the clustering termination by comparing it with a pre-defined threshold. The $\mathbf{I}$ is an indicator function having the value of 1 when the cluster size N_k satisfies $N_k > t$ (t is a threshold) or 0 otherwise. It ensures that each cluster should have a sufficient number of images for CNN training. The threshold t is set at 500 based on experiments which helps to avoid CNN overfitting.

The log probability ratio in Eq. 5 denotes the divergence of the component cluster models in the merged cluster k. In particular, the numerator corresponds to the sum of the probability that the cluster images are generated by its most probable component cluster model, and the denominator corresponds to the sum of the probability that the cluster images are generated by its most unlikely component cluster model. If the values of the two are far from each other, the component cluster models in k are considered to have large divergence and splitting should continue. Otherwise the cluster splitting terminates and the current cluster becomes a leaf cluster. During the whole cluster splitting process, the algorithm discovers the image hierarchy automatically according to the image feature similarity, where no image category label information is required.

D. SS-HCNN Training

The proposed SS-HCNN trains a hierarchy of CNNs at root, cluster and category levels to address the data annotation and uneven separability constraints. One major issue in the SS-HCNN

training is data imbalance where images in a minibatch may be routed to different clusters at different nodes. We address this issue by breaking the training into multiple phases instead of training as a whole. In particular, we first train the root node CNN which will serve as a basis for the subsequent training of cluster- and category-level CNNs at parent and leaf nodes.

The root node CNN is trained as shown in Fig. 1. With MMC-assigned cluster labels, the root CNN is fine tuned by setting the optimization objective as the cross entropy loss between the CNN-predicted cluster labels and the MMC-assigned cluster labels. Such cluster label based CNN training has two advantages. First, the base layers of the root node CNN can be shared with its child CNNs as they capture similar low-level features. The child CNNs can thus focus more on the training of their rear layers through clustering images in the corresponding clusters. Second, the computationally expensive MMC will not be required during the testing stage because the learned root-node CNN can predict a cluster label for each test image.

The cluster-level CNNs at parent nodes can inherit the base layers from their parent CNN and the rear layers are fine-tuned similarly through MMC clustering as for the root node CNN. The category-level CNN can inherit the base layers from its parent CNN but the rear layers focusing on image classification are trained by minimizing the cross entropy loss between the predicted image category label and the ground-truth image label. Note that the image category distribution in the leaf node clusters may be imbalanced which could introduce training bias. We address this issue by (i) sampling each training minibatch with as uniform category distribution as possible, and (ii) data augmentation to generate more training samples for the image categories with rare samples.

E. SS-HCNN Testing

In the testing stage, a test image is first feedforwarded to the root node where the softmax layer will output a score vector $a_k, k = 1, ..., K$ indicating the probabilities of the image belonging to the K clusters. The child clusters where the test image will be routed to is determined based on the score ratio r as follows,

$$r = \frac{\max\limits_{k=1,...,K} a_k}{\max\limits_{k=1,...,K,k\neq j} a_k} \tag{6}$$

where j refers to the child cluster having the highest score.

We use the ratio between the highest and second highest score to select the child clusters as define in Eq. 6. In particular, if the ratio r is larger than a threshold (set at 1.3 experimentally),

the cluster j has high confidence of being the right cluster and it will be selected. Otherwise, the top two clusters have high chance as the right clusters and both are selected to maximize the probability of correct routing of the test image.

The above process repeats until the test image is finally routed to one or multiple leaf node clusters. When the test image is routed to a single leaf node cluster, it is directly classified by the corresponding leaf node CNN. When the test image is routed to multiple leaf node clusters instead, it is classified by a voting strategy defined as follows,

$$y = \arg\max_{c=1,...,C} \sum_{l=1}^{L} a_c^l \tag{7}$$

where y is the determined image category for the test image, a_c^l refers to the softmax output of the l-th leaf CNN for cateogry c, C is the number of image categories and L is the number of the traversed leaf nodes by the test image. The image category that has the maximum total response across all the traversed leaf node CNNs is thus selected as the images belonging category.

IV. Experiments

We evaluate the proposed SS-HCNN on CIFAR-100[24] and ImageNet datasets [25]. CIFAR-100 is composed of 100 classes of natural images, including 50K training images and 10K testing images. ImageNet [25] consists of 1000 classes of natural images, including 1.2 million training images and 50,000 validation images. The SS-HCNN is implemented on the Caffe [26] software. The system runs on a workstation with Intel core i7-5960X CPU 3.00GHz, NVIDIA GTX-Titan GPU, and 64GB RAM.

A. CIFAR-100

Setup: The CIFAR-100 dataset is similarly pre-processed using global contrast normalization and ZCA whitening [18]. The network in network (NIN) [5] is used as the CNN structure at each node in the proposed SS-HCNN. The original NIN consists of three MLP layers. The first two MLP layers are shared between parent and child nodes. Additional layers are introduced right after the second MLP unit to make use of the local feature response. A 1000 dimensional vector is produced to represent each image for MMC clustering. All other network parameter settings, weights initialization and learning policy strictly follow the settings provided by NIN [5].

In the implemented SS-HCNN, the NIN at each node is first pre-trained on the ImageNet dataset to avoid any prior knowledge from the CIFAR-100 dataset (namely SS-HCNN_ImageNet_NIN). The cluster-level CNNs are fine tuned by using the training images of the CIFAR-100 dataset, where the image category information is ignored. The rear discriminative layers of the leaf-node CNNs are trained by using different amounts of labelled training images to study how the SS-HCNN performs with limited image annotations. The labelled training images are selected with similar proportions from each object category to balance the category distribution within each leaf node cluster. In addition, the initial learning rate of each node CNN is 0.01, and it is decreased by a factor of 10 every 10K iterations. The minibatch size is set at 256. We set K at 3 clusters at each tree node and the cluster splitting threshold at 0.3.

Experimental Results: We compare the proposed SS-HCNN_ImageNet_NIN with the the baseline NIN [5] and the state-of-the-art hierarchical deep CNN (HD-CNN) [17] which also uses NIN as the base CNN. Fig. 2 shows the Top-1 error rates, where the horizontal axis denotes different proportions of the labelled training samples that are used for the supervised training of the leaf-node CNNs. Take the 80% case as an example. It uses 80% of labelled training images of the CIFAR-100 dataset for the training of the category-level CNNs at leaf nodes and the rest 20% training images are not used. When annotations of all training images are used, the leaf-node CNN training becomes fully supervised as shown in the 100% case in the figure. It can be seen that the performance of all three methods drops when the proportion of the image annotations used decreases. On the other hand, the SS-HCNN_ImageNet_NIN trained using 60% of the labelled training images can achieve comparable error rate (32.9%) with the fully trained HD-CNN (32.6%) using all labelled training images. This shows that the proposed SS-HCNN approach can address the data annotation constraint effectively.

We also evaluate the fully supervised SS-HCNN that is trained by using all labelled training images, and compare it with other fully supervised CNN models. Table 1 shows experimental results. It can be seen that the fully trained SS-HCNN_ImageNet_NIN achieves a testing error of 30.62%, which outperforms the state-of-the-art HD-CNN [17] and DDN [18] by around 2% and 1%, respectively.

Discussion: We study several key SS-HCNN training parameters and processes that are involved in the image hierarchy generation, cluster splitting and leaf CNN voting. In particular, one key parameter is K as described in Section 3.2 which controls how many child clusters a parent cluster is split into in the hierarchical CNN tree. Another key parameter is the cluster

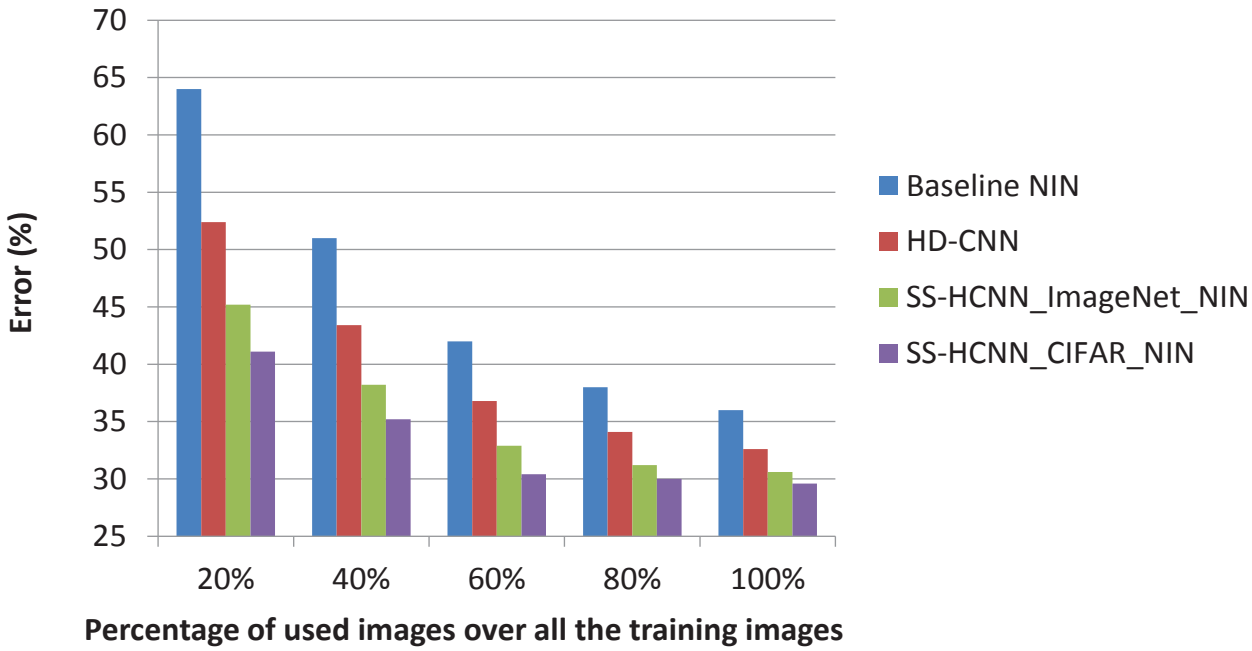

Fig. 2. Errors (%) against different proportions of labelled training images used for training in the CIFAR-100 dataset. SS-HCNN_ImageNet_NIN: SS-HCNN using ImageNet pre-trained NIN; SS-HCNN_CIFAR_NIN: SS-HCNN using CIFAR pre-trained NIN.

splitting threshold that controls the cluster splitting by comparing it with the computed metric score d as defined in Eq. 5. Beyond these two key parameters, we also design a voting based image scoring technique as described in Section 3.5 that classifies images by integrating the output of multiple leaf-node CNNs. For the clarity of presentation and ease of understanding, we study these parameters and processes by using the 60% labelled images case where the SS-HCNN achieves comparable error rate with the fully trained HD-CNN as shown in Fig. 2.

We investigate the parameters K and splitting threshold by grid search where parameter K is set at $2, 3, 4, 5, 6, 7, 8$ and meanwhile the cluster splitting threshold changes from 0.1 to 1 with a step of 0.1. Fig. 3 shows experimental results. As Fig. 3 shows, the best classification result is obtained with $K = 3$ clusters under a threshold value of 0.3. It can also be observed that a larger number of clusters require a larger threshold to achieve the optimal error rate, mainly because larger thresholds can offset the over-splitting effect as introduced by larger number of clusters. On the other hand, a smaller threshold and larger K can easily increase the over-splitting risk by splitting a cluster into a large number of child clusters. It also introduces more computations as the system needs to train a larger number of cluter-level and leaf-level CNNs. We therefore

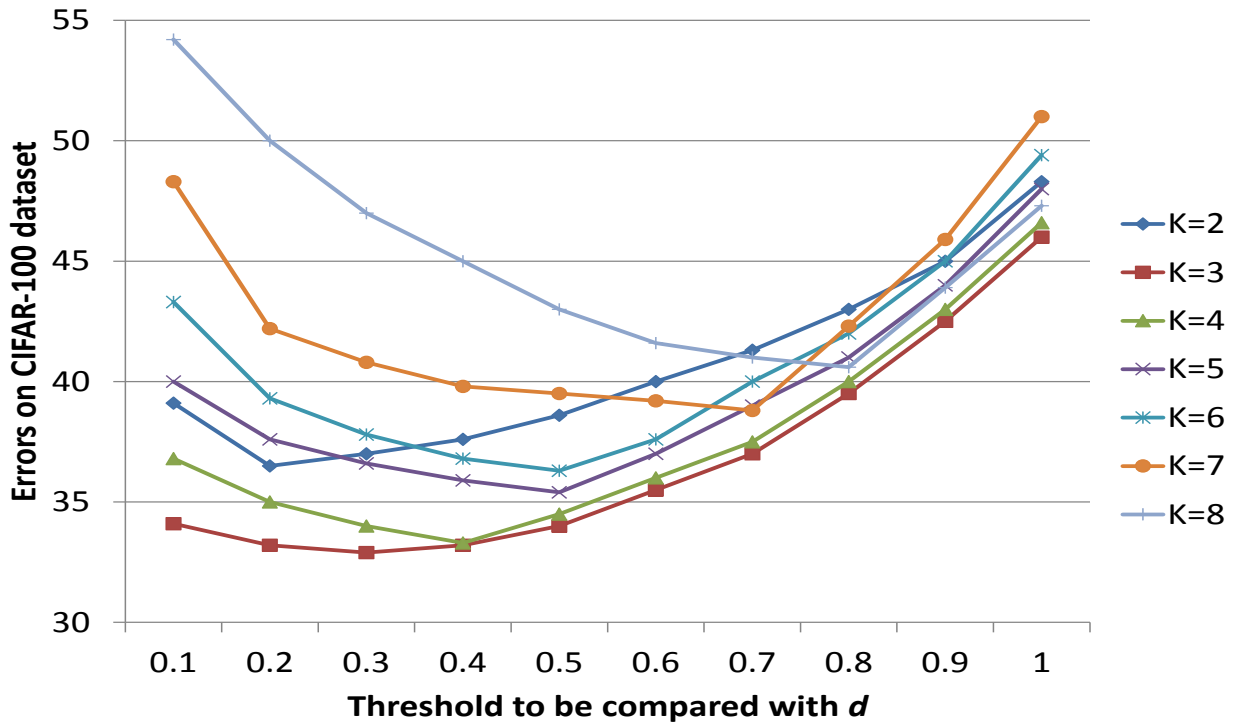

Fig. 3. Errors (%) against number of clusters and different cluster splitting thresholds on the CIFAR-100 dataset.

fix the number of clusters $K = 3$ and the cluster splitting threshold at 0.3 for the CIFAR-100 dataset as described in the previous subsection. Under this setting, we observed that there are totally 36 nodes in the tree with a depth of three layers, and the number of image categories within each leaf-node cluster ranges from 1 to 8.

We also study the voting based image scoring technique as described in Section 3.5. We compare it with the traditional hierarchical tree traversal method that determines the cluster based on the highest score only. Experimental results show that the proposed voting strategy obtains an error rate of 32.9% which is 1.1% lower than 34.0% as achieved by using the traditional tree traversal method.

Further to make a fair comparison with HD-CNN [17] as well as other state-of-the-art techniques, we also used a held-out training set (10K images as used in [17]) with label annotations from CIFAR to pre-train the NIN, and then use the pre-trained NIN to perform MMC clustering to generate coarse categories. The newly trained model is named by SS-HCNN_CIFAR_NIN as shown in Fig. 2 and Table 1. It can be seen that the SS-HCNN_CIFAR_NIN (using CIFAR pre-trained NIN) clearly outperforms the model in [17] (using CIFAR pre-trained NIN) due to

TABLE I
ERRORS (%) ON THE CIFAR-100 DATASET.

Method	Error
NIN	35.68
DSN [6]	34.68
CIFAR100-NIN	34.26
dasNet [7]	33.78
HD-CNN [17]	32.62
DDN [18]	31.65
SS-HCNN_ImageNet_NIN	30.62
SS-HCNN_CIFAR_NIN	29.64

our proposed hierarchical learning framework. At the same time, the SS-HCNN_CIFAR_NIN also outperforms the SS-HCNN_ImageNet_NIN (using ImageNet pre-trained NIN). The better performance can be explained by the CIFAR pre-trained NIN which is supervised and has better representative capability for the CIFAR images as compared with the ImageNet pretrained NIN.

B. ImageNet

Experiment Setup: For the ImageNet, we adopt the VGG-16 [2] as the network structure at each node of the SS-HCNN. The layers from conv1_1 to pool4 are shared between parent and child nodes, and the remaining layers are used as rear discriminative layers for image classification. All other network parameter settings and learning policy follow the settings provided by VGG-16. To ensure that the SS-HCNN has no prior knowledge of the ImageNet dataset, we use the VGG model pre-trained on the CIFAR-100 dataset, and then fine tune each node CNN by using images in the ImageNet dataset as described in Section 3.2 (namely SS-HCNN_CIFAR_VGG). The minibatch size is also set at 256.

Similar to the CIFAR-100 dataset, different amounts of labelled training images are employed to train the leaf-node CNNs. The image hierarchy is set with $K = 4$ clusters at each node and the cluster splitting threshold is set at 0.3. The initial learning rate for each node CNN is set at 0.001, and it is decreased by a factor of 10 every 4K iterations.

Experimental Results: We compare the SS-HCNN_CIFAR_VGG with the baseline VGG [2] and the hierarchical deep CNN (HD-CNN) [17]. Fig. 4 shows Top-1 error rates on the ImageNet validation when different amounts of labelled training images are used. As Fig. 4 shows, the

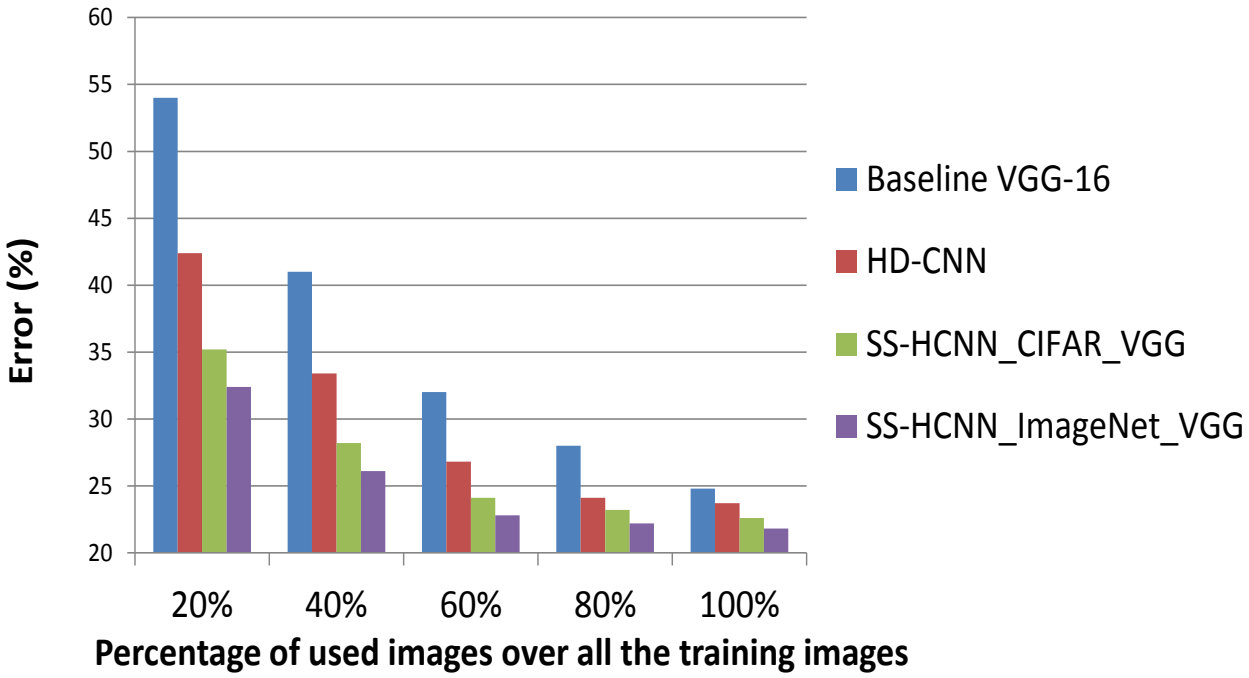

Fig. 4. Error rates (%) of different methods against different proportions of used labelled training images on the ImageNet validation set. SS-HCNN_CIFAR_VGG: SS-HCNN using CIFAR pre-trained VGG; SS-HCNN_ImageNet_VGG: SS-HCNN using ImageNet pre-trained VGG

error rates of all three methods drop as the proportion of the used image annotations increases, but the SS-HCNN_CIFAR_VGG has the least error rate drop, followed by the HD-CNN and the baseline VGG. On the other hand, it can be observed that the SS-HCNN_CIFAR_VGG trained using 60% of labelled training images can achieve comparable error rate (24.1%) with the fully trained HD-CNN (23.7%) using all labelled training images. This again demonstrates that the proposed SS-HCNN approach can address the data annotation and uneven data separability constraints effectively.

We also compare the fully supervised SS-HCNN trained using all annotated images with other CNN models including the GoogLeNet [27], baseline VGG-16 [2], VGG-19 layer network [2], dense VGG-16-layer+VGG-19-layer [17] and HD-CNN [17] which are also fully trained by using all labelled training images. Table 2 shows experimental results. It can be seen that the SS-HCNN obtains the lowest top-1 and top-5 error rates among all methods. Further, it is also observed that the SS-HCNN_ImageNet_VGG achieves better performance than SS-HCNN_CIFAR_VGG with the similar reason as discussed for the CIFAR experiments.

Discussion: Similar to the CIFAR-100 dataset, we study the cluster splitting number K, the

TABLE II

ERRORS (%) OF DIFFERENT FULLY TRAINED METHODS USING ALL THE TRAINING IMAGES ON THE IMAGENET VALIDATION SET.

Method	Top-1	Top-5
GoogLeNet	N/A	7.9
Baseline VGG-16-layer	24.79	7.50
VGG-19-layer	24.8	7.5
VGG-16-layer+VGG-19-layer	24.0	7.1
HD-CNN	23.69	6.76
SS-HCNN_CIFAR_VGG	22.6	5.7
SS-HCNN_ImageNet_VGG	21.8	4.8

splitting threshold and the voting based image scoring for the ImageNet dataset. For the clarity of presentation and ease of understanding, we similarly use the 60% labelled images case where the SS-HCNN achieves comparable error rate with the fully trained HD-CNN as shown in Fig. 4.

We first investigate the parameters K and splitting threshold by grid search where parameter K is set at $2, 3, 4, 5, 6, 7, 8$ and meanwhile the cluster splitting threshold changes from 0.1 to 1 with a step of 0.1. Fig. 5 shows experimental results. As Fig. 5 shows, the lowest error rate is obtained with $K = 4$ clusters under a threshold value of 0.3. Similar to the CIFAR-100 dataset, it can be observed that a larger number of clusters require a larger threshold to achieve the optimal error rate. On the other hand, a larger optimal cluster number K is observed on the ImageNet dataset because the ImageNet data has a larger number of image categories and also a larger number of images within each image category. We therefore fix the number of clusters K at 4 and the cluster splitting threshold to be 0.3 as described in the previous subsections. Under this setting, it is observed there are totally 136 nodes in the tree with a depth of four, and the category number in each leaf node ranges from 5 to 20.

We also study the voting based image scoring technique and compare it with the traditional hierarchical tree traversal method. Experimental results show that the proposed voting strategy obtains 24.1% error rate which is 1.3% lower than the 25.4% as achieved by the traditional hierarchical tree traversal method.

Similar to the experiments for the CIFAR dataset, we also follow the work [17] and use 100K

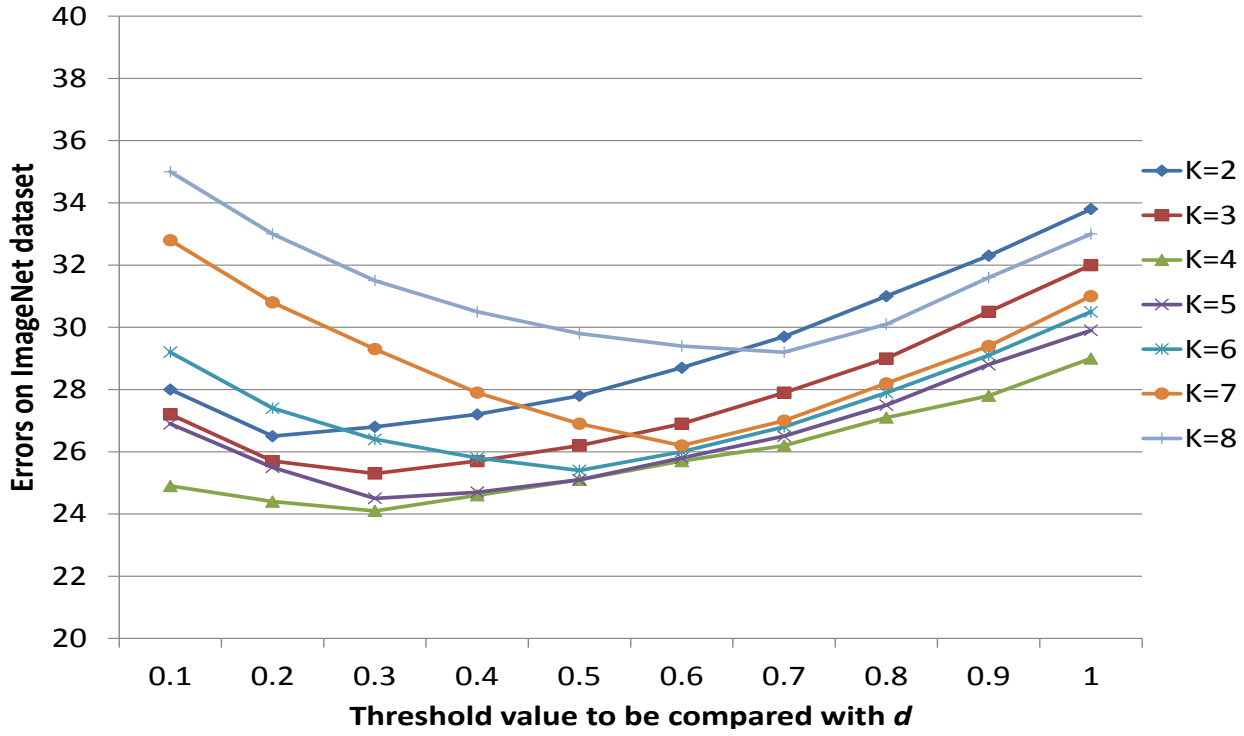

Fig. 5. Errors (%) against different numbers of clusters and cluster splitting thresholds on the ImageNet validation set.

held-out training images from the ImageNet to pre-train the VGG and then use the pre-trained VGG to perform MMC clustering to generate coarse categories. The newly trained model is named by SS-HCNN_ImageNet_VGG as shown in Table 2 and Fig. 4. It can be seen that the SS-HCNN_ImageNet_VGG clearly outperforms the model in [17] (using ImageNet pre-trained VGG) as well as the SS-HCNN_CIFAR_VGG with similar reasons.

C. *Efficiency*

We study the memory footprint and computational efficiency of the proposed SS-HCNN and compare it with the baseline CNN and the HD-CNN [17]. Table 3 shows experimental results. It can be seen that SS-HCNN consumes more memory footprint and has slightly longer testing time than the baseline and HD-CNN, largely due to its deeper hierarchy structure. In addition, HD-CNN employs product quantization for parameter compression, whereas our SS-HCNN does not perform any parameter compression. The memory footprint of the SS-HCNN can be further reduced by adopting similar parameter compression techniques.

TABLE III

COMPARISON OF MEMORY FOOTPRINT (MB) AND TESTING TIME (SECONDS) BETWEEN SS-HCNN AND OTHER NETWORKS ON CIFAR100 AND IMAGENET DATASETS. THE TESTING MINI-BATCH SIZE IS 50.

Dataset	Models	Memory	Testing time
CIFAR100	Baseline NIN	188	0.04
	HD-CNN	286	0.1
	SS-HCNN	368	0.16
ImageNet	Baseline VGG-16	4134	1.04
	HD-CNN	6863	5.28
	SS-HCNN	8672	5.68

We have also studied the Wall clock time of the SS-HCNN. For the SS-HCNN trained on ImageNet which contains 136 nodes and 5 to 20 image categories within each leaf-node cluster, it is found that fine-tuning the VGG at the root node takes around 48 hours and fine-tuning the VGG in each node of the following four levels takes an average of 11 hours, 4 hours, 2 hours and 1 hour, respectively. The whole SS-HCNN can be trained in around 3 days as the child-node VGGs at the same level can be trained in parallel and the base layers (Conv1 to Pool4) of the child-node VGGs are inherited from their parent VGG which require no further training.

For the computational complexity, it is noted that a VGG-16 model has around 15 billion flops (multiply-adds) [3]. As our SS-HCNN shares the conv1 to pool4 layers (between parent and child nodes) which takes around 88% of the total flops, the feature maps for these shared layers in the child nodes can be inherited from their parent node directly and the corresponding flops are saved accordingly. Therefore, the SS-HCNN with 136 nodes (VGGs) for the ImageNet will have around $(1-88\%)\times 135\times 15+1\times 15=258$ billion flops in the training process. During the testing stage, the proposed voting based image scoring method requires to traverse 1 (best case) or at most 2 (worst case) nodes for a parent node at each level. The flops therefore become $15\times(1+2\times 0.12+2^2\times 0.12+2^3\times 0.12+2^4\times 0.12)=69$ billion flops in the worst case, and $15\times(1+1\times 0.12+1\times 0.12+1\times 0.12+1\times 0.12)=22$ billion flops in the best case.

V. CONCLUSION

We present a semi-supervised hierarchical CNN (SS-HCNN) framework to solve the data annotation constraint and uneven data separability problem. The SS-HCNN identifies image

hierarchy using a newly designed large-scale MMC technique, and groups images into different visually compact clusters at different hierarchical levels. A stage-wise training strategy is developed to train the SS-HCNN, where cluster-level CNNs at parent nodes are first trained based on the generated cluster labels in an unsupervised manner, and category-level CNNs at leaf nodes can then be trained by using a small amount of labelled image annotations. A voting based image scoring technique is designed to classify each image. Experiments on the CIFAR-100 and ImageNet datasets show that the proposed SS-HCNN can relieve the data annotation constraint and uneven data separability challenge effectively.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[4] Z. Dong, Y. Wu, M. Pei, and Y. Jia, "Vehicle type classification using a semisupervised convolutional neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 4, pp. 2247–2256, 2015.

[5] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[6] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets.," in *AISTATS*, vol. 2, p. 6, 2015.

[7] M. F. Stollenga, J. Masci, F. Gomez, and J. Schmidhuber, "Deep networks with internal selective attention through feedback connections," in *Advances in Neural Information Processing Systems*, pp. 3545–3553, 2014.

[8] D. C. Ciresan, U. Meier, J. Masci, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, vol. 22, pp. 1237–1242, 2011.

[9] J. Weston, F. Ratle, H. Mobahi, and R. Collobert, "Deep learning via semi-supervised embedding," in *Neural Networks: Tricks of the Trade*, pp. 639–655, Springer, 2012.

[10] R. Johnson and T. Zhang, "Supervised and semi-supervised text categorization using lstm for region embeddings," in *Proceedings of The 33rd International Conference on Machine Learning*, pp. 526–534, 2016.

[11] R. Johnson and T. Zhang, "Semi-supervised convolutional neural networks for text categorization via region embedding," in *Advances in neural information processing systems*, pp. 919–927, 2015.

[12] G. Papandreou, L.-C. Chen, K. P. Murphy, and A. L. Yuille, "Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1742–1750, 2015.

[13] P. Bojanowski and A. Joulin, "Unsupervised learning by predicting noise," *arXiv preprint arXiv:1704.05310*, 2017.

[14] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam, "Large-scale object classification using label relation graphs," in *European Conference on Computer Vision*, pp. 48–64, Springer, 2014.

[15] T. Xiao, J. Zhang, K. Yang, Y. Peng, and Z. Zhang, "Error-driven incremental learning in deep convolutional neural network for large-scale image classification," in *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 177–186, ACM, 2014.

[16] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[17] Z. Yan, V. Jagadeesh, D. Decoste, W. Di, and R. Piramuthu, "Hd-cnn: hierarchical deep convolutional neural network for image classification," in *International Conference on Computer Vision (ICCV)*, vol. 2, 2015.

[18] V. N. Murthy, V. Singh, T. Chen, R. Manmatha, and D. Comaniciu, "Deep decision network for multi-class image classification," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2016.

[19] K. Ahmed, M. H. Baig, and L. Torresani, "Network of experts for large-scale image categorization," in *14th European Conference on Computer Vision*, pp. 516–532, 2016.

[20] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans, "Maximum margin clustering," in *Advances in neural information processing systems*, pp. 1537–1544, 2004.

[21] J. W. Xiao Liu, Tian Xia, "Fully convolutional attention networks for fine-grained recognition," *arXiv preprint arXiv:1603.06765*, 2016.

[22] E. Gundogdu, E. S. Parıldı, B. Solmaz, V. Yücesoy, and A. Koç, "Deep learning-based fine-grained car make/model classification for visual surveillance," in *Counterterrorism, Crime Fighting, Forensics, and Surveillance Technologies*, vol. 10441, p. 104410J, International Society for Optics and Photonics, 2017.

[23] J. Dong, Q. Chen, J. Feng, K. Jia, Z. Huang, and S. Yan, "Looking inside category: subcategory-aware object recognition," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 25, no. 8, pp. 1322–1334, 2015.

[24] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255, IEEE, 2009.

[26] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 675–678, ACM, 2014.

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