Adaptive Sharing for Image Classification

Li Shen\textsuperscript{1}, Gang Sun\textsuperscript{1,2}, Zhouchen Lin\textsuperscript{3,4,*}, Qingming Huang\textsuperscript{1,5,*}, Enhua Wu\textsuperscript{2,6}

\textsuperscript{1} University of Chinese Academy of Sciences, Beijing, China
\textsuperscript{2} State Key Lab. of Computer Science, Inst. of Software, CAS, Beijing, China
\textsuperscript{3} Key Laboratory of Machine Perception (MOE), School of EECS, Peking University, Beijing, China
\textsuperscript{4} Cooperative Medianet Innovation Center, Shanghai, China
\textsuperscript{5} Key Lab. of Intell. Info. Process., Inst. of Comput. Tech., Chinese Academy of Sciences, China
\textsuperscript{6} University of Macau, Macao, China

Abstract

In this paper, we formulate the image classification problem in a multi-task learning framework. We propose a novel method to adaptively share information among tasks (classes). Different from imposing strong assumptions or discovering specific structures, the key insight in our method is to selectively extract and exploit the shared information among classes while capturing respective disparities simultaneously. It is achieved by estimating a composite of two sets of parameters with different regularization. Besides applying it for learning classifiers on pre-computed features, we also integrate the adaptive sharing with deep neural networks, whose discriminative power can be augmented by encoding class relationship. We further develop two strategies for solving the optimization problems in the two scenarios. Empirical results demonstrate that our method can significantly improve the classification performance by transferring knowledge appropriately.

1 Introduction

Image classification is a core problem in computer vision and artificial intelligence. Learning with a number of class labels poses a great challenge on traditional multi-class classification models, where training multiple classifiers is typically independent or mutually exclusive. The task of distinguishing one class from hundreds of class labels would be difficult, especially when its training data is insufficient.

Much of the effort in deploying algorithms is devoted to leveraging rich task relationship to transfer information [Caruana, 1997; Thrun and Pratt, 1998; Evgeniou and Pontil, 2004]. The learning paradigms aim to achieve better generalization performance by encouraging common knowledge to be shared across related ones. Accordingly, the crucial aspect is the introduction of hypothesis to model the relatedness among tasks. For example, widely used assumptions that task parameters (i.e., classifier parameters) either lie in a common feature subspace [Obozinski et al., 2006; Argyriou et al., 2008; Liu et al., 2009] or share a common probabilistic prior [Yu et al., 2005; Fei-Fei et al., 2006], are essentially based on the hypothesis that all tasks are related and all relevant information should be shared. When such strong assumptions do not hold, such sharing may incur adverse effect on overall performance. To avoid this, some methods have been proposed to discover specific structures, such as outliers [Gong et al., 2012; Pu et al., 2014] or disjoint groups [Zhou et al., 2011; Kang et al., 2011; Srivastava and Salakhutdinov, 2013].

Regarding the realistic classification problems, it is complicated to model class relatedness in the target space. Determining how to share is hard to be addressed accordingly. For example, the introduction of disjoint group structure reflects the desire of intra-group sharing, which drives the classifier parameters in a group close to each other. As a matter of fact, the sharing among classes usually forms a continuum in the more realistic setting. Some classes are less related than others even if they are partitioned to a group. On the other hand, it is imperative to highlight the specific features of one class against others even if they have much in common, since the original goal of developing model is to distinguish the set of classes. A robust method is needed to effectively share information and identify individual difference.

In this regard, we propose an Adaptive Sharing method for image classification. A distinct insight from our method is to selectively share information among classes while capturing respective disparities simultaneously. The learning model is expected to leverage feature relevance when it exists, but not require it strictly satisfying certain structure. This goal is achieved by estimating a composite of two parameter sets with different types of regularization. The classifier parameters for all classes are decomposed into two parts: one corresponds to the shared features and the other corresponds to the class-specific features. A nuclear norm penalty is exploited on the first part to capture the underlying relatedness structure among classes and an element-wise sparsity penalty is imposed on the second part to highlight the disparities of each class. The objective is formulated as a non-smooth convex optimization problem when given the feature space. We...
We propose a novel Adaptive Sharing method which selectively shares information among classes and captures the class-specific properties simultaneously. Compared with imposing strong assumptions or discovering specific structures, we provide an elegant way to appropriately transfer knowledge among classes.

Besides applying it for learning classifiers on pre-computed features, we also integrate Adaptive Sharing with deep neural networks, where the performance can be improved by encoding class relatedness structure. Consequently, we develop different optimization strategies in the two scenarios. Experimental results on multiple challenging datasets demonstrate the efficacy of such selectively transfer for improving the overall classification performance.

2 Related Work

Image classification in real-world scenarios has drawn increasing attention. Complex appearance variations and class correlation bring in the difficulties for classifying many classes. Much work is proposed to study and exploit the relatedness among classes to transfer information in a multi-task learning paradigm. A family of methods are developed based on sharing a prior in the hierarchical Bayesian framework [Fei-Fei et al., 2006; Archambeau et al., 2011]. Another direction is formulating the approaches in the regularization framework where the tasks are assumed to lie in a common feature subspace [Argyriou et al., 2007; Liu et al., 2009]. However, these methods typically assume strong shared relationship among tasks, which might degrade the overall performance due to the information transfer among unrelated tasks. Some methods are further proposed to discover relatedness structure for sharing. For example, a mixed penalty is adopted in [Mei et al., 2012], outliner tasks are detected in [Gong et al., 2012; Pu et al., 2014] and task grouping is learnt in [Zhou et al., 2011; Kang et al., 2011]. Different from our method, these methods encourage the relatedness satisfying certain structural bracket. Moreover, the work in [Jalali et al., 2010; Chen et al., 2012] captures the inherent relationship among tasks while allowing the existence of different features. The two work provide theoretical analysis based on the linear feature space, whereas we are concerned with a more realistic problem and further incorporate our method with deep architectures.

Alternatively, much effort is devoted to developing feature representation learning [Krizhevsky et al., 2012; Lin et al., 2014; Lee et al., 2014; Stollenga et al., 2014; He et al., 2014; Simonyan and Zisserman, 2014], which achieves state-of-the-art performance on image classification. However, the development of these models does not take advantage of class relatedness. The work [Deng et al., 2014] exploits semantic prior in the deep model, which is different from implicitly learning the relatedness structure in our method. In [Srivastava and Salakhutdinov, 2013], a group-based structure is estimated with deep model iteratively. As any change on class partitioning would lead to the re-training of overall network, the method suffers from considerable time cost for reaching a plateau and might be intractable for dealing with many classes. Our method adopts a more concise and effective way to combine the class relationship.

3 Adaptive Sharing Approach

Assume we have a set of training images $\mathcal{X} = \{x_i\}_{i=1}^N$, $\mathcal{Y} = \{y_i\}_{i=1}^N$ is the corresponding label set. $y_i$ is a $K$ dimensional vector (whose value can be binary or one-of-$K$) for indexing target classes. The learning of each classifier is regarded as a single task. Notation $\|\cdot\|_1$, $\|\cdot\|_p$ and $\|\cdot\|_\infty$ denote the $\ell_1$ norm, Frobenius norm and nuclear norm of matrix [Lin et al., 2011], respectively. The nuclear norm is the sum of singular values.

In this section, we first describe our method based on precomputed features and the optimization strategy by using accelerated proximal gradient algorithm. Then we incorporate Adaptive Sharing with convolutional neural networks and present the corresponding optimization strategy.

3.1 Learning on pre-computed features

Suppose each image $x \in \mathcal{X}$ has been represented by a pre-computed feature vector $x \in \mathbb{R}^D$. Let $w_k \in \mathbb{R}^D$ denote the parameter vector of task $k$ (i.e., the classifier parameters of class $k$) which is a column of the parameter matrix $W \in \mathbb{R}^{D \times K}$. In the standard learning paradigm where each task is learnt independently, the objective function typically takes the form:

$$\min_{W} \frac{1}{N} \sum_{k=1}^{K} \sum_{i=1}^{N} \ell(w_k^T x_i, y_i^k) + \lambda \|W\|^2_F,$$

where $\lambda$ is the regularization factor, and $\ell(w_k^T x_i, y_i^k)$ denotes the loss between the prediction $w_k^T x_i$ and the true value $y_i^k$. When considering feature selection [Obozinski et al., 2006], $\ell_1$ norm regularization is utilized instead.

In multi-task learning paradigm, the tasks are expected to learn jointly and share a common feature subspace, such as imposing a low-rank structure [Argyriou et al., 2008]:

$$\min_{W} \frac{1}{N} \sum_{k=1}^{K} \sum_{i=1}^{N} \ell(w_k^T x_i, y_i^k) + \lambda \|W\|_\star.$$
we suppose that each task depends on the shared information and the additional specific properties.

Here we leverage a composite of two parameter vectors, $c_k$ and $s_k$, to represent the classifier parameters towards class $k$, i.e., $w_k = c_k + s_k$. The two components correspond to the shared features and the specific features, respectively. Let $C = (c_1, \ldots, c_K)$ and $S = (s_1, \ldots, s_K)$. We introduce different structures on the matrices $C$ and $S$. A low-rank structure is exploited to capture the inherent relatedness among tasks and an element-wise sparsity structure is leveraged to highlight the disparities of each task simultaneously. Formally, the learning problem can be formulated as:

$$
\min_{C, S} \frac{1}{N} \sum_{k=1}^{K} \sum_{i=1}^{N} \ell \left( (c_k + s_k)^T x_i, y_i \right) + \lambda_c \|C\|_* + \lambda_s \|S\|_1,
$$

where $\lambda_c$ and $\lambda_s$ are the penalty factors. The nuclear norm penalty has the effect of encouraging a low-rank solution on the proximal gradient algorithm [Beck and Teboulle, 2009]. Let $\mathcal{L}(\mathcal{X}, \mathcal{Y}; C, S)$ symbolically denote the empirical loss:

$$
\mathcal{L}(\mathcal{X}, \mathcal{Y}; C, S) = \frac{1}{N} \sum_{k=1}^{K} \sum_{i=1}^{N} \ell \left( (c_k + s_k)^T x_i, y_i \right),
$$

$Z$ denote the variables to be optimized,

$$
Z = \begin{pmatrix} C \\ S \end{pmatrix}, \quad C \in \mathbb{R}^{D \times K}, \quad S \in \mathbb{R}^{D \times K},
$$

$g(\cdot)$ and $h(\cdot)$ refer to the smooth term and non-smooth convex term, respectively,

$$
g(Z) = \mathcal{L}(\mathcal{X}, \mathcal{Y}; C, S), \quad h(Z) = \lambda_c \|C\|_* + \lambda_s \|S\|_1.
$$

The updating at the $t$-th iteration performs as follows:

$$
Z_{t+1} = \arg \min_{Z} \left( h(Z) + \frac{1}{2\eta_t} \|Z - (U_t - \eta_t \nabla g(U_t))\|_F^2 \right),
$$

where $\eta_t$ denotes step size. $g(Z)$ is approximated by a quadratic local model around $U_t$. The variable $U_t$ can be set as a combination of $Z_t$ and $Z_{t-1}$ from previous iterations:

$$
U_t = Z_t + b_{t-1} - 1 \frac{1}{b_t} (Z_t - Z_{t-1}),
$$

where $b_t = \left(1 + \sqrt{4b_{t-1}^2 + 1}\right)/2$ for $t \geq 1$, and $b_0 = 1$. Considering that (7) takes an equivalent form:

$$
\min_{C, S} \frac{1}{2} \|C - \hat{C}_t\|_F^2 + \frac{1}{2} \|S - \hat{S}_t\|_F^2 + \lambda_c \|C\|_* + \lambda_s \|S\|_1,
$$

where

$$
\begin{pmatrix} \hat{C}_t \\ \hat{S}_t \end{pmatrix} \triangleq U_t - \eta_t \nabla g(U_t), \quad \lambda_c = \eta_t \lambda_c, \quad \lambda_s = \eta_t \lambda_s.
$$

We can leverage the decomposability in (9) to optimize variables $C$ and $S$ separately, and the closed-form solutions can be obtained [Lin et al., 2011], respectively.

### 3.2 Integrating Adaptive Sharing with Deep Neural Networks

Deep neural networks have shown strong power on image classification. We aim to integrate Adaptive Sharing into deep neural network framework to augment the network by encoding the relatedness among classes. Our model can be implemented as a standalone layer in such a framework. We exploit it to replace the last full-connected layer in deep neural networks (such as convolutional neural networks [Krizhevsky et al., 2012]), that is to say, the last layer weight parameters (connected to $K$ nodes corresponding to the $K$ classes) are comprised of two components.

The neural network can be regarded as a feature space projection for each image $x_i$. Different from learning classifiers on pre-computed features (in Section 3.1), the parameters in the projection should be learnt jointly. Let $\theta$ denote the parameters in the network except the ones at the last layer. Then the empirical loss in (4) can be reformulated with negative log-likelihood:

$$
\mathcal{L}(\mathcal{X}, \mathcal{Y}; C, S, \theta) = -\frac{1}{N} \sum_{i=1}^{N} \log P(y_i|x_i, C, S, \theta).
$$

$\theta$ can be regarded as the parameters to generate the representation for image $x_i$. The matrices $C$ and $S$ compose the last layer weight parameters, i.e., $w_k = c_k + s_k$, which denote the weight parameters towards class $k$.

The optimization can be done using stochastic gradient descent method with mini-batches. However, the problem is non-trivial due to the low-rank constraint on the matrix $C$. Therefore, we adopt an operator $\Omega_{\varepsilon}(C)$ [Mei et al., 2012] to approximate the nuclear norm penalty $\lambda_c \|C\|_*$:

$$
\Omega_{\varepsilon}(C) = \min_{Q} \left( \frac{1}{2\varepsilon} \|Q - C\|_F^2 + \lambda_c \|Q\|_* \right),
$$

where $\varepsilon$ is the approximation factor. The approximation $\Omega_{\varepsilon}(C)$ is convex and smooth with respect to $C$. Then the gradient can be computed as:

$$
\nabla \Omega_{\varepsilon}(C) = \lambda_c (C - Q^\dagger),
$$

where $Q^\dagger = \arg \min_Q \left( \frac{1}{2\varepsilon} \|Q - C\|_F^2 + \lambda_c \|Q\|_* \right)$. With respect to nuclear norm, $Q^\dagger$ can be computed with a closed-form expression by utilizing the soft-thresholding operator on the singular values of the matrix $C$ [Lin et al., 2011]. Consequently, the gradient of the regularized loss over a batch of data $D = \{x_i, y_i\}$ is estimated as:

$$
\left\langle \frac{\partial \mathcal{L}(x_i, y_i)}{\partial C} \right\rangle_D + \nabla \Omega_{\varepsilon}(C),
$$

where $\langle \cdot \rangle_D$ denotes the average operator over the batch $D$. The update rule for the parameters $S$ and $\theta$ follows the standard algorithm [Jia et al., 2014].

### 4 Experiments

We present extensive empirical studies to evaluate our Adaptive Sharing learning in two scenarios: pre-computed features and deep neural networks. We first compare our method with

2185
some representative multi-task learning methods in the fine-grained classification problem and assess the performance on a widely used dataset CUB-200-2010 [Welinder et al., 2010]. Moreover, we evaluate it on two challenging datasets, CIFAR-100 [Krizhevsky, 2009] and ImageNet 2012 classification dataset [Russakovsky et al., 2014], in the context of deep architectures.

### 4.1 Evaluation with Pre-computed Features

The CUB-200-2010 (Birds200) is a widely used dataset for fine-grained classification (FGC). It contains 6,033 images of birds belonging to 200 species, where only 15 images per class are used for training, the rest are used for testing. Some example images from the dataset are shown in Fig. 1. Each image is first cropped against the provided bounding box and resized such that the longer side is no more than 300 pixels. Our method does not specify features, and we use kernel descriptors (KDES) [Bo et al., 2010] as the image-level representation. Specifically, four types of the KDES are applied: gradient-based, color-based, normalized color-based, and local-binary-pattern-based. The patch size is set to $16 \times 16$, and the stride is set to 8 pixels. We adopt the squared hinge loss [Yang et al., 2009]. We change $\lambda_c$ from 0 to 1.0 with step 0.1, and $\lambda_s$ from 0.001 to 1.0 with ratio 10.

We compare our Adaptive Sharing with three representative multi-task learning methods (MTL): JFS [Argyriou et al., 2007], CMTL [Zhou et al., 2011] and GMTL [Pu et al., 2014]. All the methods are based on the same KDES features as ours. The regularization parameters are chosen by cross validation on the training set for all the MTL methods. In order to comprehensively assess the results, we also provide some FGC methods as baselines. Recently, several work exploits the techniques of segmentation and localization on object parts to obtain strong performance on the dataset, such as [Chai et al., 2013; Gavves et al., 2013]. These methods contain extra procedures, and the results highly depend on the quality of segmentation and localization. Thus, we leave them out of the comparison, however our advantage is complementary to their strength of modeling object parts.

The results of different methods on overall 200 classes are summarized in Table 1. Our result is obtained when $\lambda_c = 0.8$ and $\lambda_s = 0.1$. The performance is measured by accuracy. Adaptive Sharing achieves better result than baseline methods. In particular, we can observe that it clearly outperforms baseline MTL methods. Compared with the FGC methods, baseline MTL methods achieve marginal improvement, even worse performance. Complex intra-class variations and inter-class correlation incur negative transfer in these methods, which hurts the overall performance. In contrast, our method appropriately exploits shared information by jointly capturing the task relationships and individual disparities.

To further evaluate the performance among the MTL methods, we test them on a frequently used subset of CUB-200-2010 dataset, Birds14. The subset is comprised of 14 species...
Table 2: Architectures of the networks used for classification on CIFAR-100. The convolutional layer is denoted as “conv <receptive field>, <filters>”. The max-pooling layer is denoted as “maxpool <region size>, <stride>”.

<table>
<thead>
<tr>
<th>input size</th>
<th>model A / A-AS</th>
<th>model B / B-AS</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>conv 5×5, 96</td>
<td>conv 3×3, 64</td>
</tr>
<tr>
<td></td>
<td>maxpool 3×3, 2</td>
<td>maxpool 2×2, 2</td>
</tr>
<tr>
<td>16</td>
<td>conv 5×5, 128</td>
<td>conv 3×3, 128</td>
</tr>
<tr>
<td></td>
<td>maxpool 3×3, 2</td>
<td>maxpool 2×2, 2</td>
</tr>
<tr>
<td>8</td>
<td>conv 5×5, 256</td>
<td>conv 3×3, 256</td>
</tr>
<tr>
<td></td>
<td>maxpool 3×3, 2</td>
<td>spp. {4, 2, 1}</td>
</tr>
<tr>
<td>-</td>
<td>FC, 2048</td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>FC, 2048</td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>FC, 100 / AS, 100</td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>softmax</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Performance comparison of the four models on CIFAR-100. In the brackets are the improvements over the “no Adaptive Sharing” baselines.

<table>
<thead>
<tr>
<th>Model</th>
<th>A</th>
<th>A-AS</th>
<th>B</th>
<th>B-AS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test error (%)</td>
<td>37.20</td>
<td>35.75 (1.45)</td>
<td>33.17</td>
<td>31.20 (1.97)</td>
</tr>
</tbody>
</table>

Table 4: Test errors on CIFAR-100.

<table>
<thead>
<tr>
<th>Method</th>
<th>Test error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree based Priors</td>
<td>36.85%</td>
</tr>
<tr>
<td>[Srivastava and Salakhutdinov, 2013]</td>
<td></td>
</tr>
<tr>
<td>Network in Network [Lin et al., 2014]</td>
<td>35.68%</td>
</tr>
<tr>
<td>Deeply Supervised [Lee et al., 2014]</td>
<td>34.57%</td>
</tr>
<tr>
<td>dasNet [Stollenga et al., 2014]</td>
<td>33.78%</td>
</tr>
<tr>
<td>Deeper Network (B)</td>
<td>33.17%</td>
</tr>
<tr>
<td>Adaptive Sharing (B-AS)</td>
<td>31.20%</td>
</tr>
</tbody>
</table>

Table 5: Test errors (%) of B-AS against parameter λc and ε.

<table>
<thead>
<tr>
<th>ε</th>
<th>λc</th>
<th>10^−1</th>
<th>10^−2</th>
<th>10^−3</th>
<th>10^−4</th>
</tr>
</thead>
<tbody>
<tr>
<td>10^0</td>
<td>32.45</td>
<td>31.87</td>
<td>31.39</td>
<td>32.03</td>
<td></td>
</tr>
<tr>
<td>10^−1</td>
<td>32.61</td>
<td>31.81</td>
<td>31.48</td>
<td>31.84</td>
<td></td>
</tr>
<tr>
<td>10^−2</td>
<td>32.23</td>
<td>31.82</td>
<td><strong>31.20</strong></td>
<td>31.97</td>
<td></td>
</tr>
</tbody>
</table>

4.2 Evaluation on Deep Neural Networks

The CIFAR-100 dataset [Krizhevsky, 2009] is composed of 32 × 32 color images belonging to 100 classes, with 50,000 images for training and 10,000 images for testing. We choose this dataset because there are a large number of classes but each one has a few samples, making it suitable for demonstrating the efficacy of sharing paradigm. The dataset is preprocessed by subtracting the mean image over the training set. We evaluate without any data augmentation.

Our Adaptive Sharing (AS) is independent of the deep architectures used. We investigate two different convolutional neural network architectures, as shown in Table 2. The model A is the one used in [Hinton et al., 2012]. Inspired by [Simonyan and Zisserman, 2014], we develop a deeper network, model B, by replacing one 5 × 5 convolutional (conv) layer with a stack of two 3 × 3 conv layers. We also incorporate spatial pyramid pooling (spp) [He et al., 2014] into the model B, where the pyramid configuration is {4, 2, 1}.

For the CIFAR-100 dataset [Krizhevsky, 2009], we achieve a significant improvement over the baseline methods. The clear improvement over the CMTL and GMTL results is evident. This further illuminates that our method benefits from appropriately sharing information, and hence can effectively improve the generalization performance.

Our Adaptive Sharing (B-AS) is independent of the deep architecture used. We investigate two different convolutional neural network architectures, as shown in Table 2. The model A is the one used in [Hinton et al., 2012]. Inspired by [Simonyan and Zisserman, 2014], we develop a deeper network, model B, by replacing one 5 × 5 convolutional (conv) layer with a stack of two 3 × 3 conv layers. We also incorporate spatial pyramid pooling (spp) [He et al., 2014] into the model B, where the pyramid configuration is {4, 2, 1}. All weights layers (except for the last Fully-Connected (FC) layer) are followed by the Rectified Linear Unit (ReLU).

Dropout is applied to all the pooling layers and the first two FC layers, with the dropout ratios 0.25 and 0.5, respectively. We implement the Adaptive Sharing as a standalone layer, which can be integrated by replacing the last FC layer. We denote the models as A-AS and B-AS, respectively. Consequently, there are four models, A and B, as well as A-AS and B-AS for comparison.

Our implementation is based on the publicly available code of Caffe [Jia et al., 2014]. We train the networks by applying stochastic gradient descent with a mini-batch size of 128 and a fixed momentum of 0.9. The training is regularized by weight decay (the ℓ2 penalty factor is set to 0.004). Particularly, the parameter matrix S in Adaptive Sharing is regularized with ℓ1 weight decay (the penalty factor is set to 0.0005). The learning rate is initialized to 0.001, is divided by 10 when the error plateaus.

The performance on the four models is shown in Table 3. With respect to model A, we achieve a coincide result as the one reported in [Hinton et al., 2012]. By integrating our Adaptive Sharing, the model performance can be effectively enhanced (1.45% improvement). On the other hand, [Srivastava and Salakhutdinov, 2013] achieves marginal improvement (test error is 36.85%, with 0.35% improvement) by using the same network configuration. The method is also formulated in transfer learning framework. However, information sharing is strictly governed by the determination of group structure, which may hurt the performance due to hard partitioning. In contrast, our method does not strictly require the relatedness to satisfy certain structure. It is flexible to couple related classes, and the information can be appropriately transferred in our model. The clear improvement over [Srivastava and Salakhutdinov, 2013] (1.10%) demonstrates the
advantage of Adaptive Sharing. It is worth noting that Adaptive Sharing can enhance the model A and B consistently, and the superiority is more significant in the deeper network B. This implies that such sharing paradigm is beneficial for discovering useful features.

In order to comprehensively confirm the effectiveness of our method, we compare with the previous state-of-the-art results, as shown in Table 4. By virtue of a deeper architecture, the model B (in Table 3) outperforms the published best result. However, the superiority is marginal. Our Adaptive Sharing (B-AS) further improves the result. A test error of 31.20% is achieved, which surpasses dasNet [Stollenga et al., 2014] by 2.58%.

The balance between the shared part C and the specific part $S$ can be regularized by the values of the parameters $\lambda_s$ and $\lambda_c$. Due to many parameters in deep neural networks, we apply a simple strategy that specifies $\lambda_s$ (and other parameters in the networks) with aforementioned empirical value, and study the effect of low rank parameters (i.e., the regularization factor $\lambda_c$ and the approximation factor $\varepsilon$ in (11)) on model performance. The test errors of B-AS with different values of $\lambda_s$ and $\varepsilon$ are summarized in Table 5. The results show that the model is insensitive to the factor $\varepsilon$, that similar performance can be obtained with a fixed $\lambda_s$. In contrast, the regularization factor $\lambda_c$ plays a critical role in model performance. The best result is obtained when $\lambda_c$ is set to $10^{-3}$.

To investigate the power of our Adaptive Sharing in capturing the class relatedness, we make an analysis on the matrix $C$ (corresponding to the shared features) of the model B-AS. We utilize $C^TC$ to represent the similarity matrix of all the classes in CIFAR-100, which is shown in Fig. 3. Darker color describes larger value in the similarity matrix (diagonal line denotes self-similarity). The matrix exhibits block-diagonal structure, indicating that related classes are encouraged to be coupled and share information. While hard partitioning is not applied in Adaptive Sharing, we adopt the Normalized Cut [Shi and Malik, 2000] to visualize the induced class groups of the similarity matrix, as shown in Table 6. For comparison, we also provide the class groups of the model B by adopting a similar operation on the weight matrix of the last FC layer, and the result is shown in Table 7. It is obvious that the relatedness captured in Adaptive Sharing is more reasonable. In fact, the cluster relationship among classes is introduced in Adaptive Sharing is more reasonable. In fact, the cluster relationship among classes is introduced in Adaptive Sharing is more reasonable. In fact, the cluster relationship among classes is introduced in Adaptive Sharing is more reasonable. In fact, the cluster relationship among classes is introduced in Adaptive Sharing is more reasonable. In fact, the cluster relationship among classes is introduced in Adaptive Sharing is more reasonable. In fact, the cluster relationship among classes is introduced in Adaptive Sharing is more reasonable. In fact, the cluster relationship among classes is introduced in Adaptive Sharing is more reasonable. In fact, the cluster relationship among classes is introduced in Adaptive Sharing is more reasonable. In fact, the cluster relationship among classes is introduced in Adaptive Sharing is more reasonable.

We perform additional experiments on the ImageNet 2012 classification dataset [Russakovsky et al., 2014], which is a challenging dataset with 1000 classes. We use the AlexNet architecture [Krizhevsky et al., 2012] (based on Caffe training protocol [Jia et al., 2014]) as baseline, which achieves 57.1% on top-1 accuracy and 80.2% on top-5 accuracy on the validation set, using the center crop. We integrate our Adaptive Sharing in the network by replacing the last FC layer. We apply the same setting of the parameters $\lambda_c$, $\lambda_s$ and $\varepsilon$ as in CIFAR-100, and set the other parameters the same as the baseline. Our model is trained on a single Tesla K40 GPU within two weeks. We obtain 57.7% on top-1 accuracy and 81.3% on top-5 accuracy on the validation set, where the improvements over the baseline are 0.6% and 1.1%, respectively. Due to the high baseline on top-5 accuracy, the improvement is more difficult than the one on top-1 accuracy.
5 Conclusion

In this paper, we present a novel adaptive sharing method for image classification. The shared information is selectively extracted and exploited to improve the generalization performance while simultaneously identifying the class-specific properties. We further integrate such adaptive sharing with deep neural networks. The outstanding performance on multiple challenging datasets verifies the effectiveness of such adaptive transfer. As a future direction, we are interested in leveraging such sharing paradigm to model the relation among the filters in deep neural networks.

Acknowledgments

This work was supported in part by National Basic Research Program of China (973 Program): 2012CB316400, in part by National Natural Science Foundation of China: 61332016, 61025011 and 61272326, in part by Grant of University of Macau (MYRG202(Y1-L4)-FST11-WEH). Z. Lin is supported by 973 Program of China (grant no. 2015CB352502), NSF China (grant nos. 61272341 and 61231002), and Microsoft Research Asia Collaborative Research Program. We would like to thank NVIDIA for GPU donation.

References


