Training Hierarchical Feed-Forward Visual Recognition Models Using Transfer Learning from Pseudo-Tasks

Amr Ahmed[1,⋆], Kai Yu[2], Wei Xu[2], Yihong Gong[2], and Eric Xing[1]

[1] School of Computer Science, Carnegie Mellon University
{amahmed,epxing}@cs.cmu.edu
[2] NEC Labs America, 10080 N Wolfe Road, Cupertino, CA 95014
{kyu,xw,ygong}@sv.nec-labs.com

Abstract. Building visual recognition models that adapt across different domains is a challenging task for computer vision. While feature-learning machines in the form of hierarchial feed-forward models (e.g., convolutional neural networks) showed promise in this direction, they are still difficult to train especially when few training examples are available. In this paper, we present a framework for training hierarchical feed-forward models for visual recognition, using transfer learning from pseudo tasks. These pseudo tasks are automatically constructed from data without supervision and comprise a set of simple pattern-matching operations. We show that these pseudo tasks induce an informative inverse-Wishart prior on the functional behavior of the network, offering an effective way to incorporate useful prior knowledge into the network training. In addition to being extremely simple to implement, and adaptable across different domains with little or no extra tuning, our approach achieves promising results on challenging visual recognition tasks, including object recognition, gender recognition, and ethnicity recognition.

1 Introduction

Visual recognition has proven to be a challenging task for computer vision. This difficulty stems from the large pattern variations under which an automatic recognition system must operate. Surprisingly, this task is extremely easy for humans, even when very few examples are available to the learner. This superior performance is in fact due to the expressive hierarchical representation employed by human visual cortex. Therefore, it has been widely believed that building robust invariant feature representation is a key step toward solving visual recognition problems.

In the past years, researchers have designed various features that capture different invariant aspects in the image, to name a few: shape descriptors [21], appearance descriptors like SIFT features and their variants [16], etc. A classifier is then feed with this representation to learn the decision boundaries between the object classes. On the other hand, many efforts have been put toward building *trainable* vision systems in the form of *hierarchical feed-forward models* that learn the feature extractors and the classification model simultaneously. This approach emulates processing in the visual

⋆ Work mainly done while the author was interning at NEC labs.

D. Forsyth, P. Torr, and A. Zisserman (Eds.): ECCV 2008, Part III, LNCS 5304, pp. 69–82, 2008.

cortex and is reminiscent of the Hubel-Wiesel architecture [12]. While we concede that given enough time and proper understanding of a particular visual recognition problem, researchers can devise ingenious feature extractors that would achieve excellent classification performance especially when the learner is faced with few examples, we believe that it is hard to devise a single set of features that are *universally* suitable for all recognition problems. Therefore, it is believed that learning the features automatically via biologically inspired models will open the door for more robust methods with wider applications.

In this paper, we focus on Convolutional Neural Networks (CNNs) as an example of trainable hierarchical feed-forward models [15]. CNNs have been successfully applied to a wide range of applications, including character recognition, pose estimation, face detection, and recently generic object recognitions. The model is very efficient in the recognition phase because of their feed-forward nature. However, this generality and capacity of handling a wide variety of domains comes with a price: the model needs a very large number of labeled examples per class for training. To solve this problem, recently an approach has been proposed that utilizes unlabeled data in the training process [20]. Even though the method improves the performance of the model, to date, the best reported recognition accuracy on popular benchmarks like Caltech101 by hierarchical feed-forward models are yet unsatisfactory [14].

In this paper, we present a framework for training hierarchical feed-forward models by leveraging knowledge via transfer learning from a set of pseudo tasks which are automatically constructed from data without supervision . We show that these auxiliary tasks induce a *data-dependent* inverse-Wishart prior on the parameters of the model. The resulting framework is extremely simple to implement, in fact, nothing is required beyond the ability to train a hierarchical feed-forward model via backpropagation. We show the adaptability and effectiveness of our approach on various challenging benchmarks that include the standard object recognition datasets Caltech101, gender classification, and ethnic origin recognition on face databases FERET and FRGC [19]. Overall, our approach, with minimal across-domain extra tuning, exhibits excellent classification accuracy on all of these tasks, outperforming other feed-forward models and being comparable to other state-of-the-art methods. Our results indicate that:

- Incorporation of prior knowledge via transfer learning can boost the performance of CNNs by a large margin.
- Trainable hierarchical feedforward models, have the flexibility to handle various visual recognition tasks of different nature with excellent performance.

2 Related Work

Various techniques have been proposed that exploit locally invariant feature descriptors, to name a few: appearance descriptors based on SIFT features and their derivatives [16], shape descriptors [21], etc. Based on these feature descriptors, a similarity measure is induced over images, either in the bag of word representation [8], or in a multi-resolution representation [14]. This similarity measure is then then used to train a discriminative classifier. While these approaches achieve excellent performance, we

believe that it is hard to devise a single set of features that are *universally* suitable for all visual recognition problems.

Motivated by the excellent performance and speed of the human visual recognition system, researchers explored the possibility of learning the features automatically via hierarchical feedforward models that emulate processing in the visual cortex. These approaches are reminiscent of multi-stage Hubel-Wiesel architectures that use alternating layers of convolutional feature detectors (simple cells) and local pooling and subsampling (complex cells) [12]. Examples of this generic architecture include: [7],[22],[18] in addition to Convolutional Neural Networks (CNN) [15] (see Fig. 2). Several approaches have been proposed to train these models. In [22] and [18] the first layer is hard-wired with Gabor filters, and then large number of image patches are sampled from the second layer and used as the basis of the representation which is then used to train a discriminative classifier. In CNN all the layers, including a final layer for classification, are jointly trained using the standard backpropagation algorithm [15]. While this approach makes CNN powerful machines with a capacity to adapt to various tasks, it also means that large number of training examples are required to prevent overfitting. Recently [20] proposed a layer-wise greedy algorithm that utilizes unlabeled data for pre-training CNNs. More recently, in [13], the authors proposed to train a feed-forward model jointly with an unsupervised embedding task, which also leads to improved results. Though using unlabeled data too, our work differs from the previous work at the more emphasis on leveraging the prior knowledge which suggests that our work can be combined with those approaches to further enhance the training of feed-forward models in general and CNN in particular, as we will discuss in section 4.

Finally, our work is also related to a *generic*transfer learning framework [2], which uses auxiliary tasks to learn a linear feature mapping. The work here is motivated differently and aims toward learning complex nonlinear visual feature maps as we will discuss in section 3.3. Moreover, in object recognition, transfer learning has been studied in the context of probabilistic generative models [6] and boosting [23]. In this paper our focus is on using transfer learning to train hierarchical feedforward models by leveraging information from unlabeled data.

3 Transfer Learning

3.1 Basics

Transfer learning, also known as multi-task learning [1,5], is a mechanism that improves generalization by leveraging shared domain-specific information contained in related tasks. In the setting considered in this paper, all tasks share the same input space (X) and each task m can be viewed as a function f_m that maps between this space to an output space: $f_m : X \rightarrow Y$. Intuitively, if the tasks are truly related, then there is a shared structure between all of all them that can be leveraged by learning them in parallel. For example, Fig 1-a depicts few tasks. In this figure it is clear that input points a and b[1] have similar values across all of these tasks, and thus one can conclude that these two input points are semantically similar, and therefore should be assigned similar values

[1] Please note that the order of points along the x-axis does not necessarily encode similarity.

under other related tasks. When the input space X represents images, the inclusion of related tasks would help induce similarity measures between images that enhances the generalization of the main task being learned. The nature of this similarity measure depends on the architecture of the learning system. For instance, in a feed-forward Neural Network (NN) with one hidden layer, all tasks would share the same hidden representation (feature space) $\Phi(x)$ (see Fig. 1-b) and thus the inclusion of pseudo tasks in this architecture would *hopefully* result in constraining the model to map semantically similar points like a and b ,from the input space, to nearby positions in the feature space.

3.2 Problem Formulation

Since in this paper we mainly focus on feed-forward models, we will formulate our transfer learning problem using a *generic* neural network learning architecture as in Fig. 1-b. Let N be the number of input examples, and assume that the main task to be learnt has index m with training examples $D_m = \{(x_n, y_{mn})\}$. A neural network has a natural architecture to tackle this learning problem by minimizing:

$$\min_{\theta} l\left(\mathrm{D_m}, \theta\right) + \gamma\Omega(\theta) \tag{1}$$

where $l\left(\mathrm{D_m}, \theta\right)$ amounts to an empirical loss

$$\min_{w_m} \left[\sum_{n} \ell(y_{mn}, w_m^{\mathrm{T}} \Phi(x_n; \theta)) + \alpha \|w_m\|^2 \right]$$

$\Omega(\theta)$ is a regularization term on the parameters of the feature extractors $\Phi(x;\theta) = [\phi_1(x;\theta) \ldots \phi_J(x;\theta)]^{\mathrm{T}}$ – this feature extractor, i.e. the hidden layer of the network, maps from the input space to the feature space. Moreover, $\ell_m(\cdot,\cdot)$ is the cost function for the target task. Unlike the usual practice in neural networks where the regularization on θ is similar to the one on w_m, we adopt a more informative $\Omega(\theta)$ by additionally introducing Λ *pseudo* auxiliary tasks, each represented by learning the input-output pairs: $D_\lambda = \{(x_n, y_{\lambda n})\}_{n=1}^{N}$, where $y_{\lambda n} = g_\lambda(x_n)$ are a set of real-valued functions automatically constructed from the input data. As depicted in Fig. 1.b, all the tasks share the hidden layer feature mapping. Moreover, we hypothesis that each pseudo auxiliary function, $g_\lambda(x_n)$, is linearly related to $\Phi(x_n;\theta)$ via the projection weights w_λ. Then the regularization term $\Omega(\theta)$ becomes:

$$\min_{\{w_\lambda\}} \sum_{\lambda} \left[\sum_{n} \left(y_{\lambda n} - w_\lambda^{\mathrm{T}} \Phi(x_n;\theta) \right)^2 + \beta \|w_\lambda\|^2 \right] \tag{2}$$

Training the network in 1.b to realize the objective function in (1) is extremely simple because nothing beyond the standard back-propagation algorithm is needed. By constructing meaningful pseudo functions from input data, the model is equipped with extensive flexibilities to incorporate our prior knowledge. Furthermore, there is no restriction on the parametric form of $\Phi(x;\theta)$, which allows us to apply learning problem (1) to more complex models (e.g., the CNN shown in Fig. 2.a). Our experiments will demonstrate that these advantages can greatly boost the performance of CNNs for visual recognition.

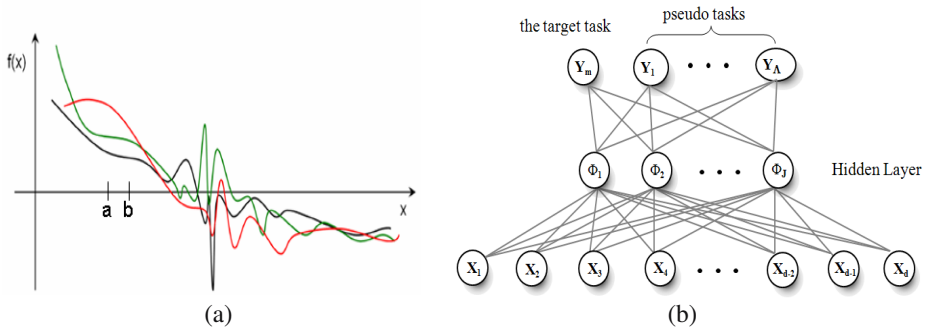

Fig. 1. Illustrating the mechanism of transfer learning. (a) Functional view: tasks represented as functional mapping share stochastic characteristics. (b) Transfer learning in neural networks, the hidden layer represents the level of sharing between all the task.

3.3 A Bayesian Perspective

In this section we give a Bayesian perspective to the transfer learning problem formulated in Section 3.2. While (1, 2) are all what is needed to implement the proposed approach, the *sole* purpose of this section is to give more *insight* to the role of the *pseudo* tasks and to formalize the claims we made near the end of Section 3.1.

In Section 3.2, we hypothesized that the pseudo tasks are realizable as a linear projection from the feature mapping layer output, $\Phi(x;\theta)$, that is:

$$y_\lambda = w_\lambda^{\mathrm{T}} \Phi(x;\theta) + e \tag{3}$$

where $e \sim \mathrm{N}(0, \beta^{-1})$. The intuition behind (3) is to limit the capacity of this mapping so that the constraints imposed by the pseudo tasks can only be satisfied by proper adjustments of the feature extraction layer paraments, θ. To make this point more clear, consider Fig. 1.a, and consider points like a and b which are assigned similar values under many pseudo tasks. Under the restriction that the pseudo auxiliary tasks are realizable as a linear projection from the feature extraction layer output, and given an appropriate number of such pseudo tasks, the only way that the NN can satisfy these requirements, is to map points like a and b to nearby position in the feature space. Therefore, the kernel induced by the NN, $K(x_i, x_j;\theta)$, via its feature mapping function $\Phi(.;\theta)$, is constrained to be similar to the kernel induced by the pseudo tasks, where the degree of similarity is controlled via the parameter γ in (1). Below we will make this intuition explicit.

We first begin by writing the empirical loss due to the pseudo auxiliary tasks, $\mathrm{L}(\{D_\lambda\}, \theta, \{w_\lambda\})$, where we make the dependency on $\{w_\lambda\}$ explicit, as follows:

$$\mathrm{L}(\{D_\lambda\}, \theta, \{w_\lambda\}) = \sum_\lambda \left[\sum_n \left(y_{\lambda n} - w_\lambda^{\mathrm{T}} \Phi(x_n;\theta) \right)^2 + \beta \|w_\lambda\|^2 \right] \tag{4}$$

If we assumer that $w_\lambda \sim \mathrm{N}(0, \mathrm{I})$, and that $e \sim \mathrm{N}(0, \beta^{-1})$, then it is clear that (4) is the negative log-likelihood of $\{D_\lambda\}$ under these *mild* Gaussian noise assumptions.

In Section 3.2, we decided to *minimize* this loss over $\{w_\lambda\}$, which gives rise to the regularizer term, $\Omega(\theta)$. Here, we will take another approach, and rather *integrate* out $\{w_\lambda\}$ from (4), which results in the following fully Bayesian regularizer, $\Omega_B(\theta)$:

$$\Omega_B(\theta) = \frac{\Lambda}{2}\log\det(\Phi^{\mathrm{T}}\Phi + \beta^{-1}\mathrm{I}) + \frac{\Lambda}{2}\mathrm{tr}\Big((\Phi^{\mathrm{T}}\Phi + \beta^{-1}\mathrm{I})^{-1}\mathrm{K}_\Lambda\Big) \tag{5}$$

where $K_\Lambda = \frac{\sum_{\lambda=1}^{\Lambda} K_\lambda}{\Lambda}$ and $K_\lambda = [g_\lambda(x_i)g_\lambda(x_j)]_{i,j=1}^{N}$. If we let $K(\theta)$ denotes the Kernel induced by the NN feature mapping layer, where $K(x_i, x_j, \theta) = \langle \Phi(x_i;\theta), \Phi(x_j;\theta)\rangle + \delta_{ij}\beta^{-1}$, then (5) can be written as:

$$\Omega_B(\theta) = \frac{\Lambda}{2}\log\det(\mathrm{K}(\theta)) + \frac{\Lambda}{2}\mathrm{tr}\Big(\mathrm{K}(\theta)^{-1}\mathrm{K}_\Lambda\Big) \tag{6}$$

It is quite easy to show that (6) is equivalent to a loss term due to an *inverse-wishart* prior, $\mathrm{IW}(\Lambda, \mathrm{K}_\Lambda)$, placed over $K(\theta)$. Put it another way, (6) is the KL-divergence between two multivariate normal distributions with zero means and covariance matrices given by $K(\theta)$ and K_Λ. Therefore, in order to minimize this loss term the leaner is biased to make the kernel induced by the NN,$K(\theta)$, as similar as possible to the kernel induced by the pseudo-tasks,K_Λ, and this helps regularize the functional behavior of the network, especially when there are few training examples available. In Section 3.2, we choose to use the regularizer, $\Omega(\theta)$ as a proxy for $\Omega_B(\theta)$ for efficiency as it is amenable to efficient integration with the online stochastic gradient descent algorithm used to train the NN, whereas $\Omega_B(\theta)$ requires gradient computations over the whole pseudo auxiliary task data sets, for every step of the online stochastic gradient algorithm. This decision turns out to be a sensible one, and results in an excellent performance as will be demonstrated in Section 6.

4 Transfer Learning in CNNs

There are no constraints on the form of the feature extractors $\Phi(.;\theta)$ nor on how they are parameterized given θ, therefore, our approach is applicable to any feed-forward architecture as long as $\Phi(.;\theta)$ is differentiable, which is required to train the whole model via backpropagation. A popular architecture that showed excellent performance for visual recognition is the CNN architecture, see Fig. 2.a, which is an instance of multi-stage Hubel-Wiesel architectures [12],[15]. The model includes alternating layers of convolutional feature detectors (C layers), and local pooling of feature maps using a max or an averaging operation (P layers), and a final classification layer. Detailed descriptions of CNNs can be found in [15]. Applying the transfer learning framework described in Section 3 to CNNs results in the architecture in Fig. 2-a. The pseudo tasks are extracted as described in Section 5 and the whole resulting architecture is then trained using standard backpropagation to minimize the the objective function in (1).

Throughout the experiments of this paper, we applied CNNs with the following architecture: (1) Input: 140x140 pixel images, including R/G/B channels and additionally two channels Dx and Dy, which are the horizontal and vertical gradients of gray intensities; (2) C1 layer: 16 filters of size 16×16; (3) P1 layer: max pooling over each

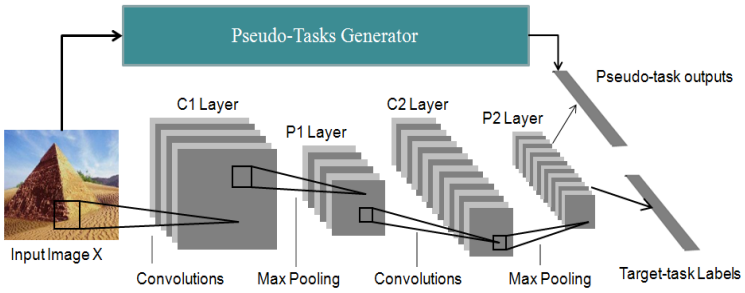

Fig. 2. Joint training using transfer-learning from pseudo-tasks

5×5 neighborhood; (4) C2 layer: 256 filters of size 6×6, connections with sparsity[2] 0.5 between the 16 dimensions of P1 layer and the 256 dimensions of C2 layer; (5) P2 layer: max pooling over each 5×5 neighborhood; (6) output layer: full connections between $256 \times 4 \times 4$ P2 features and outputs. Moreover, we used least square loss for pseudo tasks and *hinge loss* for classification tasks. Every convolution filter is a linear function followed by a sigmoid transformation (see [15] for more details).

It is interesting to contrast our approach with the layer-wise training one in [20]. In [20], each feature extraction layer is trained to model its input in a layer-wise fashion: the first layer is trained on the raw images and then used to produce the input to the second feature extraction layer. The whole resulting architecture is then used as a multilayered feature extractor over labeled data, and the resulting representation is then used to feed an SVM classifier. On contrast, in our approach, we *jointly* train the classifier and the feature extraction layers, thus the feature extraction layer training is guided by the pseudo-tasks as well as the labeled information simultaneously. Moreover, we believe that the two approaches are orthogonal as we might first pre-train the network using the method in [20], and then use the result as a starting point for our method. We leave this exploration for future work.

5 Generating Pseudo Tasks

We use a set of pseudo tasks to incorporate prior knowledge into the training of recognition models. Therefore, these tasks need to be 1) automatically computable based on unlabeled images, and 2) relevant to the specific recognition task at hand, in other words, it is highly likely that two semantically similar images would be assigned similar outputs under a pseudo task.

A simple approach to construct pseudo tasks is depicted in Fig. 4. In this figure, the pseudo-task is constructed by sampling a random 2D patch and using it as a template to form a local 2D filter that operates on every training image. The value assigned to an image under this task is taken to be the maximum over the result of this 2D convolution operation. Following this method, one can construct as many pseudo-tasks as required.

[2] In other words, on average, each filter in C2 is connected to a randomly chosen 8 dimensions (filter maps) from P1.

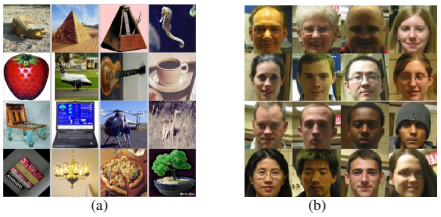

Fig. 3. Images from: (a)Caltech101 and (b)FRGC 2.0

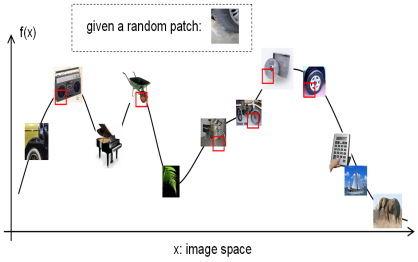

Fig. 4. Simple pseudo task generation

Moreover, this construction satisfies condition (2) above as semantically similar images are likely to have similar appearance. Unfortunately, this simple construction is brittle with respect to scale, translation, and slight intensity variations, due to operating directly on the pixel-level of the image. Below, we show how to generalize this simple approach to achieve *mild* local-invariance with respect to scale, translation and slight intensity variations.

First, we processed all the images using a set of Gabor filters with 4 orientations and 16 scales. This step aims toward focusing the pseudo-tasks on *interesting* parts of the images by using our *prior* knowledge in the form of a set of Gabor filters. Then a max-pooling operation, across scale and space, is employed to achieve *mild* scale and translation-invariance. We then apply the simple method detailed above to this representation. It is interesting to note that this construction is similar in part to [22] which used random patches as the parameters of feed-forward filters which is later used as the basis for the representation. The detailed procedure is as follows, assuming each image is a 140×140 gray image: (1) Applying Gabor filters result in 64 feature maps of size 104×104 for each image; (2) Max-pooling operation is performed first within each non-overlapping 4×4 neighborhood and then within each band of two successive scales resulting in 32 feature maps of size 26×26 for each image; (3) An set of K RBF filter of size 7×7 with 4 orientations are then sampled and used as the parameters of the pseudo-tasks. To generate the actual values of a given pseudo-task, we first process each training image as above, and then convolve the resulting representation with this pseudo-task's RBF filter. This results in 8 feature maps of size 20×20; Finally, max pooling is performed on the result across all the scales and within every non-overlapping 10×10 neighborhood, giving a 2×2 feature map which constitutes the value of this image under this pseudo-task. Note that in the last step instead of using a global max-pooling operator over the whole image, we maintained some 2D spatial information by this local max operator, which means that the pseudo-tasks are 4-dimensional vector-valued functions, or equivalently, we obtained $4 * K$ pseudo-tasks (K actual random patches, each operating at a different quadrant of the image).

These pseudo-tasks encode our prior knowledge that a similarity matching between an image and a spatial pattern should tolerate a small change of scale and translation as well as slight intensity variation. Thus, we can use these functions as pseudo tasks to train our recognition models. We note that the framework can generally benefit from

all kinds of pseudo task constructions that comply with our prior knowledge for the recognition task at hand. We have tried other ways like using histogram features of spatial pyramid based on SIFT descriptors and achieved a similar level of accuracy. Due to space limitation, we only report the results using the method detailed in this section.

6 Experimental Results

To demonstrate the ability of our framework to adapt across domains with little tuning, first, we fixed the architecture of CNN as descried in Section 4. Second, we fixed the number of pseudo tasks K = 1024. To speed up the training phase, we apply PCA to reduce these resulting pseudo-tasks to 300 ones. Moreover, in order to ensure that the neural network is trained with balanced outputs, we further project these 300 dimensions using a random set of 300 orthonormal bases and scale each of the response dimensions to have a unitary variance.

6.1 Object Recognition

We conducted experiments on the Caltech-101 database, which contains 102 categories (including 101 object categories plus a background category) of object images, with from 31 to 800 images per category. We chose Caltech-101, because the data set is considered one of the most diverse object databases available today, and more importantly, is probably the most commonly tested benchmark in the literature of object recognition, which makes our results directly comparable with those of others. We follow the standard setting in the literature, namely, train on 15/30 images per class and test on the rest. For efficiency, we limit the number of test images to 30 per class. Note that, because some categories are very small, we may end up with less than 30 test images. To reduce the overweight of popular categories, we first compute the accuracy within each category and then compute the average over all the categories. All the experiments were randomly repeated for 5 trails.

Table 1. Categorization accuracy of different hierarchical feed-forward models on Caltech-101

Training Size	15	30
HMAX-1 [22]	35%	42%
HMAX-2 [18]	51%	56%
CNN + Pretraining [20]	-	54%
CNN	23.9%	25.1%
CNN+Transfer	**58.1**%	**67.2**%

Table 1 shows the comparison of our results with those reported in the literature using similar hierarchical feed-forward models on the same settings of experiments. The baseline method "CNN", i.e., CNN trained without pseudo tasks, presented very poor accuracy, which is close to the phenomenon observed in [20]. The "CNN+Pretraining" approach made a significant improvement by first training a encoder-decoder architecture with unlabeled data, and then feeding the result of applying the encoder on labeled data to an SVM classifer [20]. The idea was inspired by [11] that suggested an unsupervised layer-wise training to improve the performance of deep belief networks. Our strategy "CNN+Pseudo Tasks" also improved the baseline CNN by a large margin, and achieved

the best results of hierarchical feedforward architectures on the Caltech 101 data set. To better understand the difference made by transfer learning with pseudo tasks, we visualize the learnt first-layer filters of CNNs in Fig. 5 (a) and (b). Due to lacking of sufficient supervision in such a high-complexity learning task, a bit surprisingly, CNN cannot learn any meaningful filters. In contrast, thanks to the additional bits of information offered by pseudo tasks, CNN ends up with much better filters. Our result is comparable to the state-of-the-art accuracy, i.e., $64.6\% \sim 67.6\%$ in the case of 30 training images per class, achieved by the spatial pyramid matching (SPM) kernel based on SIFT features [14][9]. However, the feedforward architecture of CNN can be more efficient in recognition phase. In our experiments, it takes in average 0.18 second in a PC with 2.66 GHz CPU, to process one 140×140 color image, including feature extraction and classification.

6.2 Gender and Ethnicity Recognition

In this section we work on gender and ethnicity recognitions based on facial appearance. We use the FRGC 2.0 (Face Recognition Grand Challenge[19]) data set, which contains 568 individuals' face images under various lighting conditions and backgrounds, presenting in total 14714 face images. Beside person identities, each image is annotated with gender, age, race, as well as positions of eyes and nose. Each face image is aligned based on the location of eyes, and normalized to be with zero mean and unitary length. We note that the data set is not suitable for research on age prediction, because majority of individuals are young students.

We built models for binary gender classification and 3-class ethnicity recognition, i.e., classifying images into "white", "asian", and "other". For comparison, we implemented two state-of-the-art algorithms that both utilize *holistic* facial information: one is "SVM+SPM", namely, the SVM classifier using SPM kernels based on dense SIFT descriptors, as described by [14]; the other is "SVM+RBF", namely, the SVM classifier using radius basis function (RBF) kernels operating directly on the aligned face images. The second approach has demonstrated state-of-the-art accuracy for gender recognition [3,17]. We fix 114 persons' 3014 images (randomly chosen) as the testing set, and train the recognition models with various randomly selected 5%, 10%, 20%, 50%, and "All" of the remaining data, in order to examine the model's performance given different training sizes. Note that we strictly ensure that a particular individual appear only in the test set or training set. For each training size, we randomize the training data 5 times and report the average error rate as well as the standard deviation. The results are shown in Table 2 and Table 3.

Table 2. Error of gender recognition on the FRGC data set

Training Size	5%	10%	20%	50%	All
RBF+SVM	$16.7 \pm 2.4\%$	$13.4 \pm 2.4\%$	$11.3 \pm 1.0\%$	$9.1 \pm 0.5\%$	8.6%
SPM+SVM	$\mathbf{15.3 \pm 2.9}\%$	$12.3 \pm 1.1\%$	$11.1 \pm 0.6\%$	$10.3 \pm 0.8\%$	8.7%
CNN	$61.5 \pm 7.3\%$	$17.2 \pm 4.3\%$	$8.4 \pm 0.5\%$	$6.6 \pm 0.3\%$	5.9%
CNN+Transfer	$16.9 \pm 2.0\%$	$\mathbf{7.6 \pm 1.1}\%$	$\mathbf{5.8 \pm 0.3}\%$	$\mathbf{5.1 \pm 0.2}\%$	$\mathbf{4.6}\%$

Table 3. Error of ethnicity recognition on the FRGC data set

Training Size	5%	10%	20%	50%	All
RBF+SVM	$22.9 \pm 4.7\%$	$16.9 \pm 2.3\%$	$14.1 \pm 2.2\%$	$11.3 \pm 1.0\%$	10.2%
SPM+SVM	$23.7 \pm 3.2\%$	$22.7 \pm 3.6\%$	$18.0 \pm 3.6\%$	$15.8 \pm 0.7\%$	14.1%
CNN	$30.0 \pm 5.1\%$	$13.9 \pm 2.4\%$	$10.0 \pm 1.0\%$	$8.2 \pm 0.6\%$	6.3%
CNN+Transfer	**16.0** $\pm$ **1.7**%	**9.2** $\pm$ **0.6**%	**7.9** $\pm$ **0.4**%	**6.4** $\pm$ **0.3**%	**6.1**%

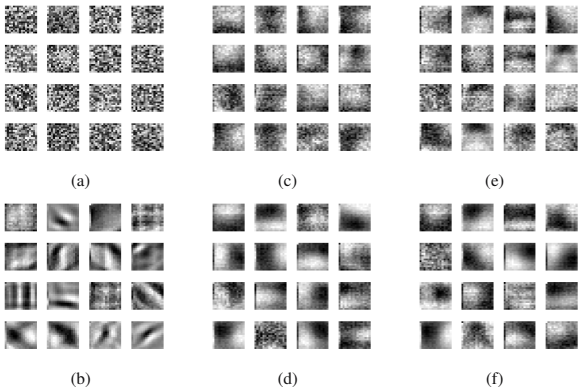

Fig. 5. First-layer filters on the B channel, learnt from both supervised CNN and CNN with transfer Learning. **top**: filters learnt from supervised CNN. **bottom**: filters learnt using transfer learning from pseudo-tasks. **first column:** Caltech-101 (30 examples per class); **second column:** FRGC-gender; and **third column:** FRGC-Race.

From Table 2 and 3 we have the following observations: (1) The two competitor methods resulted in comparable results for gender classification, while for ethnicity recognition SVM+RBF is more accurate than SVM+SPM; (2) In general, CNN models outperformed the two competitors for both gender and ethnicity recognition, especially when sufficient training data were given; (3) CNN without transfer learning produced very poor results when only 5% of the total training data were provided; (4) "CNN+Transfer" significantly boosted the recognition accuracy in nearly all the cases. In cases of small training sets, the improvement was dramatic. In the end, our methods achieved 4.6% error rate for gender recognition and 6.1% for ethnicity recognition.

Interestingly, although CNN and "CNN+Transfer" resulted in close performances when all the training data were employed, the filters learnt by CNN+Transfer (visualized in Fig. 5.d appear to be much smoother than those learnt by CNN (shown in Fig. 5.c[3] Moreover, as indicated by Fig. 6, we also found that "CNN+Transfer" converged much faster than CNN during the stochastic gradient training, indicating another advantage of our approach.

We note that the best performances our method achieved here are not directly comparable to those reported in [10,17], because their results are based on the FERET data

[3] To save space, here we only show the filters of one channel for gender and ethnicity recognition. However the same phenomenon was observed for filters of other channels.

Table 4. Error of gender recognition on the FERET data set

	RBF+SVM	Boosting	CNN	CNN+Transfer
Error	6.5%[3]	5.6%[3]	2.3%	**1.7**%

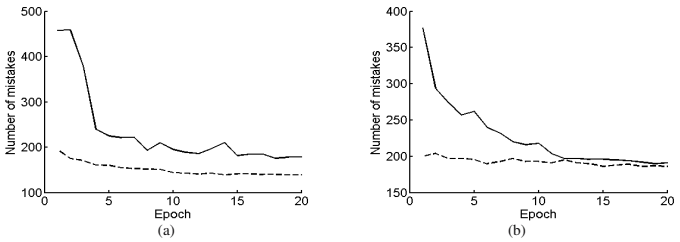

Fig. 6. Number of errors on test data over epochs, where dashed lines are results of CNN with transfer learning, solid lines are CNN without transfer learning: (a) gender recognition; (b) ethnic recognition

set[4], which contains face images under highly controlled lighting conditions and simpler backgrounds. More importantly, as recently pointed by [3], their experiments mixed up faces of same individuals across training and test sets, which made the results not truly measuring the generalization performance of handling new individuals. To make a direct comparison possible, we followed the experimental setting of [3] as much as possible, and conducted experiments on the FERET data for gender recognition, where no individual is allowed to appear in the training and test simultaneously. The results are summarized in Table 4, showing that"CNN+Transfer" achieved the best accuracy on the FERET data set.

6.3 A Further Understanding of Our Approach

In the previous two subsections we showed that our framework, with little tuning, can adapt across different domains with favorable performance. It is interesting to isolate the source of this success. Is it *only* because of the informativeness of the pseudo-tasks used? And if not, then is there a simpler way of combining the information from the pseudo-tasks with its equivalent from a supervised CNN trained only on labeled data?

To answer the first question, as we mentioned in Section 5, our pseudo-task construction overlaps with the the system in [22],[18] [5], however, our results in Table 1 indicates significant improvement over these baseline. To answer the second question, we did an additional experiment on Caltech101, using 30 training examples per category, to train an SVM on the features produced by the pseudo-tasks alone or on the combined features produced by the pseudo-tasks and the features from the last layer of a CNN trained via purely supervised learning. The results were 49.6% and 50.6% respectively. This shows that the gain from using the features from a supervised CNN was minimal. On the other

[4] Available at http://www.itl.nist.gov/iad/humanid/feret/

[5] In fact, the system in [22] and its successor [18] has other major features like inhibition,etc.

hand, our approach which involves joint-training of the whole CNN inherits the knowledge from the pseudo-tasks in the form of its induced kernel, as explained in Section 3.3, but is also supervised by labeled data and thus has the ability to further adapts its induced kernel, $K(\theta)$, to better suit the task at hand.

Moreover, our approach results in an efficient model at prediction time. In fact, the pseudo-task extraction phase is computationally expensive and it took around 29 times longer to process one image than a feedforward pass over the final trained CNN. In other words, we paid some overhead in the training phase to compute these pseudo-tasks once, but created a fast, compact, and accurate model for prediction.

7 Discussion, Conclusion, and Future Work

Benefiting from a deep understanding of a problem, hand-engineered features usually demonstrate excellent performances. This success is in a large sense due to the fact that the features are *learnt* by the smartest computational units – brains of researchers. In this sense, hand-craft designing and automatic learning of visual features do not have fundamental differences. An important indication of this paper is that, it is generally hard to build a set of features that are universally suitable for all different tasks. For example, the SPM kernel based on SIFT is excellent for object recognition, but may not be good for gender and ethnicity recognition. Interestingly, an automatically learnable architecture like CNN can adapt itself to a range of situations and learn significantly different features for object recognition and gender recognition (if comparing Fig. 5 (b) and (d)). We believe that given a sufficient amount of time, very likely researchers can come up with even better features for any visual recognition task. However, a completely trainable architecture can hopefully achieve good results for a less well-studied task with minimum human efforts.

In this paper, we empirically observed that training a hierarchical feedforward architecture was extremely difficult. We conjecture that the poor performance of CNN on Caltech 101 is due to the lack of training data, given the large variation of object patterns. In the tasks of gender and ethnicity recognitions, where we have sufficient data, CNNs in fact produced poor results on small training sets but excellent results given enough training data (see Table 2 and Table 3). Therefore, when insufficient labeled examples are present, it is essential to use additional information to supervise the network training.

We proposed using transfer learning to improve the training of hierarchical feedforward models. The approach has been implemented on CNNs, and demonstrated excellent performances on a range of visual recognition tasks. Our experiments showed that transfer learning with pseudo tasks substantially improves the quality of CNNs by incorporating useful prior knowledge. Our approach can be combined with the pre-training strategy [20][11], which remains an interesting future work.

Very recently, [4] showed that detecting region of interest (ROI) can greatly boost the performance of SPM kernel on Caltech 101. Our work is at the level of [14] that builds classifier based on the whole image. In the future, it is highly interesting to develop a mechanism of *attention* in CNNs that can automatically focus on the most interesting region of images.

References

1. Abu-Mostafa, Y.: Learning from hints in neural networks. Journal of Complexity 6, 192–198 (1990)
2. Ando, R., Zhang, T.: A framework for learning predictive structures from multiple tasks and unlabeled data. JMLR 6, 1817–1853 (2005)
3. Baluja, S., Rowley, H.: Boosting sex identification performance. International Journal of Computer Vision (2007)
4. Bosch, A., Zisserman, A., Munoz, X.: Image classification using random forests and ferns. In: ICCV 2007 (2008)
5. Caruana, R.: Multitask learning. Machine learning. Machine Learning 28(1), 41–75 (1997)
6. Fei-Fei, L.: Knowledge transfer in learning to recognize visual object classes. In: International Conference on Development and Learning (ICDL) (2006)
7. Fukushima, K., Miyake, S.: Object recognition with features inspired by visual cortex. Pattern Recognition (1982)
8. Grauman, K., Darrell, T.: The pyramid match kernel: Discriminative classification with sets of image features. In: CVPR 2005 (2005)
9. Griffin, G., Holub, A., Perona, P.: Caltech 256 object category dataset. California Institute of Technology 04-1366 (2007)
10. Gutta, S., Huang, J., Jonathon, P., Wechsler, H.: Mixture ofo experts for classification of gender, ethnic origin, and pose of human faces. IEEE Transactions on Neural Networks (2000)
11. Hinton, G., Osindero, S., Teh, Y.-W.: A fast learning algorithm for deep belief nets. Neural Computation 18, 1527–1554 (2006)
12. Hubel, D.H., Wiesel, T.N.: Receptive fields, binocular interaction, interaction and functional architecture in the cat's visual cortex. J. Physiology 160, 106–154 (1968)
13. Weston, R.C.J., Ratle, F.: Deep learning via semi-supervised embedding. In: ICML (2008)
14. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: CVPR 2006 (2006)
15. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)
16. Lowe, D.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2) (2004)
17. Moghaddam, B., Yang, M.-H.: Learning gender with support faces. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) (2002)
18. Mutch, J., Lowe, D.G.: Multiclass object recognition with sparse, localized features. In: CVPR 2006 (2006)
19. Philips, P.J., Flynn, P.J., Scruggs, T., Bower, K.W., Worek, W.: Preliminary face recognition grand challenge results. In: Proceedings of the Sevethn International Conference on Automatic Face and Gesture Recgonition (2006)
20. Ranzato, M., Huang, F.-J., Boureau, Y.-L., LeCun, Y.: Unsupervised learning of invariant feature hierarchies with applications to object recognition. In: CVPR 2007 (2007)
21. Belongie, J.M.S., Puzicha, J.: Shape matching and object recognition using shape contexts. IEEE Trans. Pattern Anal. Mach. Intell. 24(4), 509–522 (2002)
22. Serre, T., Wolf, L., Poggio, T.: Object recognition with features inspired by visual cortex. In: CVPR 2005 (2005)
23. Torralba, A., Murphy, K., Freeman, W.: Sharing visual features for multiclass and multiview object detection. IEEE PAMI (2007)