Facial Expression Recognition via a Boosted Deep Belief Network

Ping Liu* Shizhong Han* Zibo Meng Yan Tong
Department of Computer Science and Engineering
University of South Carolina, Columbia, SC 29208
{liu264, han38, mengz, tongy}@email.sc.edu

Abstract

A training process for facial expression recognition is usually performed sequentially in three individual stages: feature learning, feature selection, and classifier construction. Extensive empirical studies are needed to search for an optimal combination of feature representation, feature set, and classifier to achieve good recognition performance.

This paper presents a novel Boosted Deep Belief Network (BDBN) for performing the three training stages iteratively in a unified loopy framework. Through the proposed BDBN framework, a set of features, which is effective to characterize expression-related facial appearance/shape changes, can be learned and selected to form a boosted strong classifier in a statistical way. As learning continues, the strong classifier is improved iteratively and more importantly, the discriminative capabilities of selected features are strengthened as well according to their relative importance to the strong classifier via a joint fine-tune process in the BDBN framework. Extensive experiments on two public databases showed that the BDBN framework yielded dramatic improvements in facial expression analysis.

1. Introduction

Facial behavior is one of the most important cues for sensing human emotion and intention in people. Driven by recent advances in human-centered computing, an automatic system for accurate and reliable facial expression analysis has emerging applications such as interactive games, online/remote education, entertainment, and intelligent transportation systems.

Facial expression analysis usually employs a three-stage training consisting of *feature learning*, *feature selection*, and *classifier construction*. First, features that capture expression related facial appearance/geometry changes are extracted from images or video sequences. These features can be either hand-designed [32, 33, 34, 25, 1, 27, 28, 26, 22, 9, 4, 10, 13] or learned from training images [6, 29, 15, 16, 36, 2, 35, 30]. Then, a subset of the extracted features, which is the most effective to distinguish one expression from the others, is selected to facilitate an efficient classification and enhance the generalization capability [1, 27]. Finally, a classifier is constructed given the extracted feature set for each target facial expression.

In the current practice of facial expression analysis, these three stages are often performed sequentially and individually. To achieve satisfactory recognition performance, extensive empirical studies are needed to search for an optimal combination of feature representation, feature set, and classifier. For a new data set, this nontrivial process usually would be repeated. Although each stage is optimized given the results from the previous stage, it lacks a feedback from the latter one. Recently, it has been demonstrated that expression recognition can benefit from performing two stages together. In one example, given predefined feature representations such as Gabor features, *feature selection* and *classifier construction* were conducted iteratively in training a boosted classifier, where a feature was selected according to the current classification error and linearly combined with previously selected features to form a strong classifier [1]. In another example, *feature learning* and *classifier construction* were performed back and forth in a Deep Belief Network (DBN) [20, 21], where a hierarchical feature representation and a logistic regression function for classification were learned alternatively.

Motivated by this, we propose a novel Boosted Deep Belief Network (BDBN) to perform the three stages in a unified loopy framework. Through the proposed BDBN framework, a set of features, which is effective to characterize expression-related facial appearance/shape changes and thus, highly discriminative for classification, can be learned and selected to form a boosted strong classifier in a statistical way. Specifically, we develop and employ a novel objective function, which accounts for recognition performance of both the strong classifier and weak classifiers (features), to drive a feature fine-tuning process. As learning continues, the strong classifier is improved and more importantly, the discriminative capabilities of selected features are strengthened according to their relative importance to the strong classifier, thanks to a *joint fine-tune process* in

*means equal contributions

the BDBN framework. As shown in Fig. 1, recognition performance of the strong classifier increases with the learning going on, and so does each selected weak classifier, i.e., a patch-based feature. In addition, much fewer features are employed at the end of training because of the improved discriminative capability.

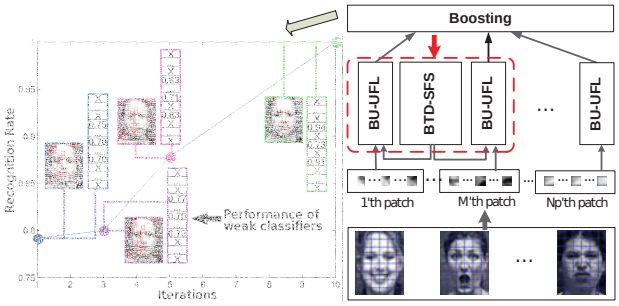


Figure 1. A boosted deep learning framework for facial expression recognition. For image patches extracted at a specific location, an initial feature representation is learned through a BU-UFL process. Then, a subset of weak learners (features enclosed in the red rectangle) is selected by boosting and fine-tuned jointly in a BTD-SFS process. The two processes run alternatively until converge. With the learning going on, the discriminative ability of the strong classifier and the weak learners increases (see the figure on the left and details can be found in Fig. 5). Best viewed in color.

As shown in Fig. 1, the BDBN framework consists of two interconnected learning processes: a *bottom-up unsupervised feature learning* (BU-UFL) process that learns hierarchical feature representations given input data and a *boosted top-down supervised feature strengthen* (BTD-SFS) process that refines the features jointly in a supervised manner. At the beginning, each training image is divided into a set of partially overlapped image patches. Next, for each set of patches extracted at the same location, an initial feature representation is learned individually in a BU-UFL process. Then, a subset of features (patches) with higher discriminative power is selected and combined to form a strong classifier in a supervised manner by boosting. The classification error from the strong classifier and from the weak classifiers (features) will be utilized and propagated backward to initiate a BTD-SFS process, where only the features selected previously would be fine-tuned *jointly* according to their contributions to minimizing an objective function. The BU-UFL and the BTD-SFS processes are iterated alternatively in a loop until converge.

Our proposed BDBN-based facial expression recognition framework has three major contributions.

- First, to the best of our knowledge, it is the first time to systematically unify feature learning, feature selection, and classifier construction in one framework.
- Second, unlike the traditional DBNs that employed the whole facial region as input [24, 20, 21], the proposed work facilitates a part-based representation, which is especially suitable for facial expression analysis.
- Third, we propose a novel discriminative deep learning framework, where the boosting technique and multiple DBNs are integrated through a novel objective function. Furthermore, the features are jointly fine-tuned such that the discriminative capability of each feature is strengthened according to its contribution to the strong classifier.

Extensive experiments on the Extended Cohn-Kanade (CK+) database [11, 17] and JAFFE database [18] showed that the BDBN framework yielded dramatic improvements in facial expression recognition compared to the state-of-the-art techniques. In addition, due to the improvement of the discriminative ability in selected features as iteration goes, the learned strong classifier only employed a few features, which demonstrated the effectiveness of feature learning/strengthen by using the proposed framework.

2. Previous Work

Extensive efforts have been devoted to recognize facial expressions [19, 31]. Facial expression recognition usually consists of two major procedures: offline training and online recognition. Generally, the system training includes three stages, i.e., feature learning, feature selection, and classifier construction.

In the first stage, features are extracted from either static images or video data to characterize facial appearance/geometry changes caused by activation of a target expression. Most of the existing work utilizes various human-crafted features including Gabor wavelet coefficients [33, 32, 25, 1, 27], Haar features [27, 28], histograms of Local Binary Patterns (LBP) [34, 26, 22, 10], Histograms of Oriented Gradients (HOG) [9, 4], scale-invariant feature transform (SIFT) descriptors [9], and 3D shape parameters [13].

Recently, unsupervised feature learning approaches especially those based on sparse-coding [6, 29, 15, 16, 36, 2, 35, 30] have been employed to extract underlying "edge-like" features from facial images and have shown promise in facial expression analysis. To become more adaptable to the real world that consists of combination of edges, deep learning networks have been employed for the applications of facial expression recognition [20, 21]. Since the whole face region is employed as input [20, 21], every part of the face is treated and fine-tuned equally no matter if it is relevant to the target facial expression.

As suggested by the psychological studies, the information extracted around nose, eyes, and mouth is more critical for facial expression analysis [3]. Furthermore, different sets of facial muscles may be involved in different facial expressions. Therefore, in the second stage, a subset of features, which is the most effective to distinguish one expression from the others, is often selected to improve the recognition performance [1, 27, 36]. For example, Zhong et al. [36] developed a two-stage multi-task sparse learning model to find common and specific facial patches, discrim-

inative to all expression categories and a target expression, respectively.

In the final stage, the extracted feature set is fed into a pre-specified classifier to train a facial expression recognizer for a target expression.

In summary, most of the aforementioned approaches perform the three training stages sequentially and individually, except for a few combining two stages [1, 27, 36, 20, 21]. Although each stage is optimized given the results from the previous one, it lacks a feedback from the latter stage. In addition, exhaustive search is needed to find an optimal combination of feature representation, feature set, and classifier given a specific dataset. In contrast, our work aims to systematically integrate the three stages in a loopy framework to yield an optimal solution consisting of a concise yet powerful feature set and a strong classifier to distinguish one facial expression from the others.

3. Boosted Deep Belief Network for Facial Expression Recognition

3.1. Overview of the BDBN Framework

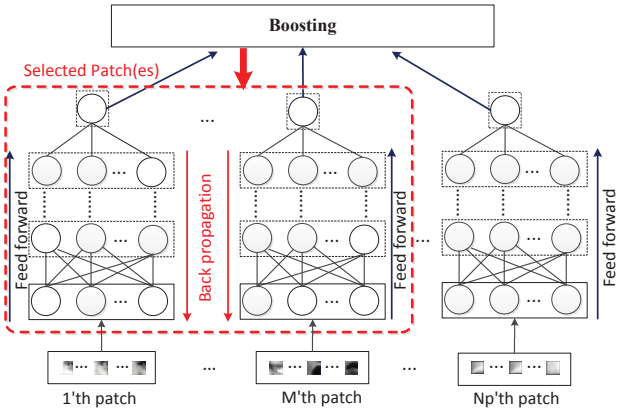


Figure 2. A BDBN consists of multiple DBNs, each of which is composed of multiple layers and intends to learn a hierarchical feature representation corresponding to image patches extracted at a specific location. Only the DBNs enclosed in the red rectangle corresponding to the patches selected by boosting will be fine-tuned jointly.

In this work, we develop a BDBN framework to perform feature learning, feature selection, and classifier construction in a loopy process. As shown in Fig. 2, the BDBN framework consists of a group of DBN structures, each of which is a multi-layer graphical model. Given a training set of image patches extracted at the same position, each DBN is utilized to learn a hierarchical feature representation. More importantly, these DBNs are connected through a boosted classifier, i.e., an AdaBoost-like classifier in this work, and fine-tuned jointly driven by a single objective function so that the features extracted at different locations are selected and strengthened jointly according to their relative importance to facial expression recognition.

The BDBN learning consists of two interconnected learning processes: a BU-UFL process and a BTD-SFS process. A BU-UFL process starts from the lowest layer and outputs in the highest layer. The feature computed in the highest layer of each DBN is employed as a weak learner to construct an AdaBoost strong classifier. The most discriminative features are selected by AdaBoost with weights proportional to their classification errors. Then, the classification errors including the overall error produced by the boosted strong classifier and the individual errors produced by the weak learners are employed to drive a BTD-SFS process, where the classification errors are back propagated to the lower levels of the DBNs. Through the BTD-SFS process, the features learned previously are fine-tuned jointly to minimize the classification errors on the training set. The BDBN learning repeats until it converges. In the subsequent discussion, we will first introduce the construction and initialization of the BDBN framework, and then present a novel BTD-SFS process for joint feature fine-tuning.

3.2. BDBN Framework Construction and Initialization based on a Group of DBNs

As shown in Fig. 2, we employ a DBN as the building block for constructing the BDBN framework. Rather than employing one DBN to learn features from the whole facial region, we divide the facial region into partially overlapped patches, each of which corresponds to a DBN, respectively. The patch-based feature representation is especially suitable for facial expression analysis as validated by the previous studies [36]. Furthermore, it facilitates a feature selection process that chooses the patches containing the most critical information of a target expression.

DBN is a hierarchical graphical model composed of layers of nodes. The nodes in the higher layer learn the statistical dependencies among the nodes in adjacent lower layer. And thus, the higher layer intends to discover more complex patterns of the input signal. Specifically, we use a DBN composed of one visual layer (the lowest layer) and five hidden layers to learn a hierarchical feature representation given training data extracted at the same patch location.

The conditional dependencies between each pair of connected layers, except the top two layers, are modeled by a Restricted Boltzmann Machine (RBM) [8]. The RBM is a two-layer undirected graphical model composed of a visible-unit layer and a hidden-unit layer. Hence, for an L+1-layer DBN, the joint distribution of the visual layer (the lowest layer) and the upper L hidden layers can be modeled as

$$\mathrm{Prob}(\mathbf{H}^0,\mathbf{H}^1,\cdots,\mathbf{H}^L)=\prod_{l=0}^{L-2}\mathrm{Prob}(\mathbf{H}^l|\mathbf{H}^{l+1})\mathrm{Prob}(\mathbf{H}^{L-1},\mathbf{H}^L) \quad (1)$$

where $\mathbf{H}^l$ denotes a set of random variables in the l^{th} layer; and $\mathbf{H}^0$ is actually the visual layer. Specifically, $\mathrm{Prob}(\mathbf{H}^{l+1}|\mathbf{H}^l)$ and $\mathrm{Prob}(\mathbf{H}^l|\mathbf{H}^{l+1})$ for $l \in [0, L-2]$ can be calculated as Eq. 2 and Eq. 3, respectively.

$$\mathrm{Prob}(\mathbf{H}^{l+1}|\mathbf{H}^l) = \frac{1}{1+\exp -(\mathbf{W}^{l,l+1}\mathbf{H}^l+\mathbf{b}_h^{l+1})} \quad (2)$$

$$\text{Prob}(\mathbf{H}^l|\mathbf{H}^{l+1}) = \frac{1}{1+\exp-\left[(\mathbf{W}^{l,l+1})^T\mathbf{H}^{l+1}+\mathbf{b}_v^l\right]} \quad (3)$$

where $\mathbf{W}^{l,l+1}$ denotes the weight matrix between the l^{th} and the $(l+1)^{th}$ layers; $\mathbf{b}_h^{l+1}$ represents the hidden bias vector at the $(l+1)^{th}$ layer; and $\mathbf{b}_v^l$ represents the visual bias vector at the l^{th} layer, respectively.

The output of DBN at the highest layer ($\mathbf{H}^L$), which represents the probability of a target expression being activated, can be estimated as

$$\mathbf{H}^L = \mathbf{W}^{L-1,L}\mathbf{H}^{L-1} \quad (4)$$

where $\mathbf{W}^{L-1,L}$ denotes the weight matrix between the top two layers.

DBN learning is to estimate $\mathbf{W}^{l,l+1}$, $\mathbf{b}_h^{l+1}$, and $\mathbf{b}_v^l$ for $l \in [0, L-2]$ as well as $\mathbf{W}^{L-1,L}$, given training data. At the beginning, an initial estimate of the parameters ($\mathbf{W}^{l,l+1}$, $\mathbf{b}_h^{l+1}$, and $\mathbf{b}_v^l$ for $l \in [0, L-2]$) can be computed using an unsupervised bottom-up learning strategy [7]. Then, through a bottom-up feed forward process, we can compute $\mathbf{H}^{L-1}$ given the input and the parameters ($\mathbf{W}^{L-2,L-1}$, $\mathbf{b}_h^{L-1}$, and $\mathbf{b}_v^{L-2}$). Given $\mathbf{H}^{L-1}$, i.e., the input from the $(L-1)^{th}$ layer, and the expression labels, the weight matrix $\mathbf{W}^{L-1,L}$ is initialized as the projection matrix of Linear Discriminant Analysis (LDA) in this work so that the output of DBN, i.e., $\mathbf{H}^L$, is a discriminative feature for facial expression recognition. Note that, this bottom-up learning process is performed for each patch location individually to learn an initial feature representation.

3.3. Joint Feature Fine-tuning in a BTD-SFS

After constructing a BDBN framework with initialized DBNs, each image patch can be transformed into a hierarchical feature representation. Then, we need to refine the features to strengthen their recognition ability by fine-tuning the DBN parameters ($\mathbf{W}$, $\mathbf{b}_h$, and $\mathbf{b}_v$). This fine-tuning process is conducted in a top-down manner by updating the weight matrix $\mathbf{W}^{L-1,L}$ first. In this work, we develop a novel BTD-SFS process, where *the fine-tuning for all DBNs is performed jointly*.

As discussed above, a BDBN framework consists of multiple DBNs. The output of each DBN at its highest layer ($\mathbf{H}^L$) can be used as a weak classifier for constructing an AdaBoost classifier. Therefore, for an image set containing N_I samples, the overall predication error is:

$$\varepsilon_{\text{strong}} = \sum_{i=1}^{N_I}\beta_i\left[\frac{1}{1+exp(-\sum_{j=1}^{M}\alpha_j \text{sgn}(\mathbf{W}_j^{L-1,L}\mathbf{H}_{i,j}^{L-1}-T_j))} - E_i\right]^2 \quad (5)$$

where $E_i \in \{0,1\}$ is the expression label of the i^{th} image; $\mathbf{W}_j^{L-1,L}\mathbf{H}_{i,j}^{L-1} = \mathbf{H}_{i,j}^L$ is the DBN output for the j^{th} selected image patch in the i^{th} image; $\boldsymbol{\alpha}$ and $\mathbf{T}$ are weights and thresholds of M selected weak classifiers in the AdaBoost classifier. $\text{sgn}(\cdot)$ is a sign function defined as:

$$\text{sgn}(\mathbf{W}_j^{L-1,L}\mathbf{H}_{i,j}^{L-1} - T_j) = \begin{cases} 1 & \text{if } \mathbf{W}_j^{L-1,L}\mathbf{H}_{i,j}^{L-1} >= T_j \\ -1 & \text{otherwise} \end{cases} \quad (6)$$

To facilitate the calculation of partial derivative of the $\text{sgn}(\cdot)$ function, we compute

$$\text{sgn}(\mathbf{W}_j^{L-1,L}\mathbf{H}_{i,j}^{L-1} - T_j) \approx \frac{\mathbf{W}_j^{L-1,L}\mathbf{H}_{i,j}^{L-1} - T_j}{\sqrt{(\mathbf{W}_j^{L-1,L}\mathbf{H}_{i,j}^{L-1} - T_j)^2 + \eta^2}} \quad (7)$$

where η is a constant to control the slope of $\text{sgn}(\cdot)$ function.

Since the number of negative samples (i.e., the images without the target expression activated) is much higher than that of positive samples (i.e., the images with the target expression), a weighting coefficient β_i is introduced to balance the contributions of the negative samples and the positive samples in Eq. 5.

Furthermore, **unlike** the traditional AdaBoost classifier, which only considers the overall classification error of the strong classifier, we propose a novel objective function that accounts for both the overall classification error $\varepsilon_{\text{strong}}$ and individual classification errors from all selected weak learners $\varepsilon_{\text{weak}}$ as follows:

$$\varepsilon = \lambda\varepsilon_{\text{strong}} + \varepsilon_{\text{weak}}, \quad (8)$$

where

$$\varepsilon_{\text{weak}} = \sum_{j=1}^{M}\alpha_j\sum_{i=1}^{N_I}\beta_i\left[\frac{\text{sgn}(\mathbf{W}_j^{L-1,L}\mathbf{H}_{i,j}^{L-1} - T_j)+1}{2} - E_i\right]^2 \quad (9)$$

and λ is a weight balancing the two terms [1].

Hence, for the k^{th} selected feature (image patch), the weight matrix between the two top layers ($\mathbf{W}_k^{L-1,L}$) can be updated by minimizing Eq. 8. In this work, we perform a line search method to search for a descent direction as:

$$\begin{aligned}\frac{\partial\varepsilon}{\partial\mathbf{W}_k^{L-1,L}} = &-2\lambda\sum_{i=1}^{N_I}\beta_i\left[\frac{1-E_i(1+A_i)}{(1+A_i)^3}\right]\frac{\partial A_i}{\partial\mathbf{W}_k^{L-1,L}} \\ &+ 2\alpha_k\sum_{i=1}^{N_I}\beta_i\left(\frac{B_{ik}+1}{2} - E_i\right)\frac{\partial B_{ik}}{\partial\mathbf{W}_k^{L-1,L}},\end{aligned} \quad (10)$$

where

$$\begin{aligned}&\frac{\partial A_i}{\partial\mathbf{W}_k^{L-1,L}} = -A_i\alpha_k\frac{\partial B_{i,k}}{\partial\mathbf{W}_k^{L-1,L}}, \frac{\partial B_{i,k}}{\partial\mathbf{W}_k^{L-1,L}} = \frac{\eta^2\mathbf{H}_{i,k}^{L-1}}{(C_{i,k}^2+\eta^2)^{\frac{3}{2}}}, \\ &A_i = \exp\left(-\sum_{j=1}^{M}\alpha_j B_{i,j}\right), B_{i,j} = \frac{C_{i,j}}{\sqrt{C_{i,j}^2+\eta^2}}, \\ &\text{and } C_{i,j} = \mathbf{W}_j^{L-1,L}\mathbf{H}_{i,j}^{L-1} - T_j.\end{aligned} \quad (11)$$

Then, the weight matrix $\mathbf{W}_k^{L-1,L}$ for the k^{th} selected feature is updated by $\mathbf{W}_k^{L-1,L} \leftarrow \mathbf{W}_k^{L-1,L} - \gamma\frac{\partial\varepsilon}{\partial\mathbf{W}_k^{L-1,L}}$, where γ is a learning rate [2].

After that, the parameters of the lower layers ($\mathbf{W}^{l,l+1}$, $\mathbf{b}_v^l$, and $\mathbf{b}_h^{l+1}$ for $l \in [0, L-2]$) are updated based on a standard back-propagation algorithm [8]. Updating parameters of the lower layers will be affected by boosting in two ways: 1) by the weighted errors estimated in boosting, and 2) by the weight matrix $W^{L-1,L}$ updated in the previous

[1] In our experiment, λ was set to 1.0 empirically.

[2] γ was initially set to 1.0 and decreased during learning in our experiments.

Algorithm 1 Iterative feature learning, feature selection, and classifier construction through a BDBN

Input: N_I training images with the corresponding expression labels $\mathbf{E}$, and the number of hidden layers of the DBN L

Output: the DBN parameters (weight matrices $\mathbf{W}^{l,l+1}$ for $l \in [0, L-1]$, visual bias $\mathbf{b}_v^l$ and hidden bias $\mathbf{b}_h^{l+1}$ for $l \in [0, L-2]$) and the AdaBoost parameters (weights $\boldsymbol{\alpha}$ and thresholds $\mathbf{T}$ for M weak classifiers)

Preprocessing: Extract N_P patches with the patch size $u \times u$ for each input image and form a set of $N_I \times N_P$ patches $\mathbf{P}$

Initialization:

for $j = 1$ to N_P **do**
  Compute $\mathbf{W}_j^{l,l+1}$, $\mathbf{b}_{v,j}^l$, and $\mathbf{b}_{h,j}^{l+1}$ for $l \in [0, L-2]$;
  Calculate $\mathbf{H}_j^{L-1}$ by contrastive divergence learning [7];
  Initialize $\mathbf{W}_j^{L-1,L}$
end for
repeat
  for $i = 1$ to N_I **do**
    for $j = 1$ to N_P **do**
      Calculate $\mathbf{H}_{i,j}^L = \mathbf{W}_j^{L-1,L}\,\mathbf{H}_{i,j}^{L-1}$;
    end for
    Form $\mathbf{H}_i^L = [\mathbf{H}_{i,1}^L, \cdots, \mathbf{H}_{i,N_P}^L]$ for the i^{th} image;
  end for
  Given $\mathbf{H}^L$ for N_I images, train an AdaBoost classifier to estimate its parameters $\boldsymbol{\alpha}$ (weights) and $\mathbf{T}$ (thresholds) for M weak classifiers
  for $k = 1$ to M **do**
    Calculate $\frac{\partial \varepsilon}{\partial \mathbf{W}_k^{L-1,L}}$ based on Eq. 10
    Update $\mathbf{W}_k^{L-1,L} \leftarrow \mathbf{W}_k^{L-1,L} - \gamma \frac{\partial \varepsilon}{\partial \mathbf{W}_k^{L-1,L}}$; {$\gamma$ is a learning rate}
    Update the parameters $\mathbf{W}_k^{l,l+1}$, $\mathbf{b}_{v,k}^l$, and $\mathbf{b}_{h,k}^{l+1}$ for $l \in [0, L-2]$ based on a standard back-propagation algorithm [8];
    Update $\mathbf{H}_k^{L-1}$.
  end for
until Converge

iteration. Note that, only the weight matrices of the selected M features would be updated, which will decrease the computation cost significantly. Then, the bottom-up and the top-down learning processes will alternatively run until converge. An algorithm for feature learning/strengthen, feature selection, and classifier construction through the BDBN framework is summarized in Algorithm **1**.

4. Experimental Results

4.1. Image Database and Experimental Setup

To demonstrate the effectiveness of the proposed BDBN framework, we have performed extensive experiments on two well-known facial expression databases: Extended Cohn-Kanade (CK+) database [11, 17] and JAFFE database [18], which have been widely used for evaluating facial expression recognition systems.

For preprocessing purpose, the face regions across different facial images were aligned given the detected eye positions to remove the scale and positional variance and then cropped to $\mathbf{167 \times 137}$. Then $\mathbf{80}$ partially overlapped image patches with a size of $\mathbf{24 \times 24}$ were extracted from each cropped facial image. For each DBN module in the BDBN framework, we employed five hidden layers plus one visual layer following the implementation of [8][3]. The number of nodes in each hidden layer is 1, 1000, 1000, 500, and 500, from the highest layer to the lowest one respectively; and the number of nodes in the visual layer is 576, which is consistent with the image patch dimension. This preprocessing strategy has been adopted in both data sets we employed.

4.2. Experiments on the CK+ Database

The CK+ database [11, 17] contains $\mathbf{327}$ expression-labeled image sequences, each of which has one of 7 expressions, i.e., anger, contempt, disgust, fear, happiness, sadness, and surprise activated. For each image sequence, only the last frame (the peak frame) is provided with an expression label. To collect more image samples from the database, we selected the last three frames for training/testing purpose from each image sequence. In addition, we also collected the first frame from each of the $\mathbf{327}$ labeled sequences for "neutral" expression. This way, an experimental data set named *CK-DB* with a total of $\mathbf{1308}$ images is built. The CK-DB was divided into 8 subsets, where the subjects in any two of subsets are not overlapped. For each run, 7 subsets were employed for training and the remaining one subset for testing. We performed such 8 runs by enumerating the subset used for testing; and the recognition performance was computed as the average of the 8 runs. In addition, an one-versus-all classification strategy was adopted to train a binary classifier for each expression.

4.2.1 Performance Evaluation on the CK-DB

We first compared the proposed BDBN framework with three baseline feature learning methods based on traditional DBNs. The first method, denoted as *GDBN*, takes the whole facial image as input to a single DBN. Each facial image is further scaled to a size of $\mathbf{24 \times 24}$ to reduce the computation complexity[4]. The comparison between *BDBN* and *GDBN* is used to demonstrate the superiority of the patch-based representation over the holistic feature representation.

The second and third methods employed the exact same input as *BDBN* (i.e., 80 image patches) and 80 DBNs, each of which corresponds to a patch location. The second method, denoted as *Ada+BUs*, learned the features by only bottom-up feature learning; while the third method, denoted as *Ada+IDBNs*, employed both bottom-up and top-down feature learning in each DBN individually (versus a joint fine-tuning in the BDBN). For both *Ada+BUs* and *Ada+IDBNs*, the outputs from the highest layers of all 80 DBNs were employed as features to train an AdaBoost classifier. The comparison between *BDBN* and

[3] We add one more hidden layer since introduction of new layers in the deep structure generally can improve the model [7].

[4] We followed the implementation in [8] for GDBN, where $\mathbf{24 \times 24}$ whole facial regions are used as input. The recognition performance on $\mathbf{48 \times 48}$ is similar to that on $\mathbf{24 \times 24}$ but with a much higher computational cost.

Ada+BUs/Ada+IDBNs intends to demonstrate the effectiveness of the joint feature learning, feature selection, and classifier construction.

For all baseline methods, we employed the traditional DBN implementation in [8] with a five-hidden-layer structure, where a two-node soft-max output layer (the highest layer) was used. Thus, the numbers of nodes are $\mathbf{2, 1000, 1000, 500, 500}$, from the highest hidden layer to the lowest one, respectively; and the numbers of nodes in the visual layer is $\mathbf{24 \times 24 = 576}$.

As shown in Fig. 3, the proposed BDBN framework outperformed all baseline methods impressively in terms of the average classification rate (**0.967**), the average hit rate (**0.891**), the average false positive rate (**0.025**) and the average F1 score (**0.834**) of the 6 basic expressions, i.e., anger, disgust, fear, happiness, sadness, surprise [5].

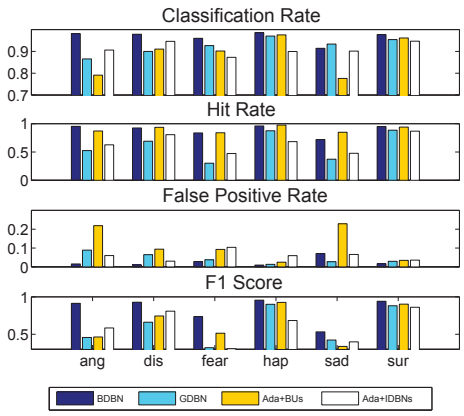


Figure 3. From top to bottom, performance comparison on the CK-DB in terms of a) classification rate, b) hit rate, c) false positive rate, and d) F1 score for 6 basic expressions. Best viewed in color.

Furthermore, we compared the proposed BDBN method with the state-of-the-art methods evaluated on CK+ or the original Cohn-Kanade database [11] [6] including methods employing LBP features [36, 23] and Gabor wavelet features [1]. To make a fair comparison, we only compared with the methods with a similar experimental setting: the last frame [1] or the last 3 frames [36, 23] in each image sequence were employed for training/testing. Among the compared methods, Common and Specific Patches (CSPL) method [36] employed multi-task learning; and AdaGabor [1] employed an AdaBoost, for feature selection, respectively. For these methods in comparison, we used their experimental results reported in their papers. As shown in Table 1, BDBN framework outperformed all the methods in comparison [1, 36, 23]. This demonstrated that the features learned and selected through BDBN contain more discriminative information for facial expression recognition.

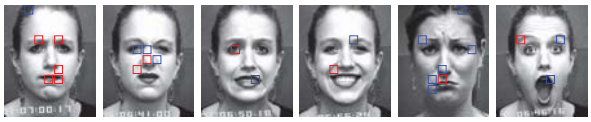
Table 1. Performance comparison on the CK+ database in terms of average classification rate for 6 expressions. LOSO: leave-one-subject-out.

Methods	CSPL [36]	AdaGabor [1]	LBPSVM [23]	**BDBN**
Validation Setting	10-Fold	LOSO	10-Fold	8-Fold
Performance	0.899	0.933	0.951	**0.967**

4.2.2 Analysis of Patches Selection Results on the CK+ Database

We are curious about what information each selected patch provides. For each expression, patches selected by the final strong classifiers through BDBN learning are marked by boxes in Fig. 4. In addition, we only show those patches that were selected more frequently in the 8-fold experiments. Specifically, patches enclosed in red boxes were selected in all the 8 runs across different subjects. These patches, we believe, contain the most discriminative information to recognize the corresponding expression. Those patches enclosed in blue boxes were selected in more than 4 runs.

Most of the selected patches, especially those enclosed in red boxes, are located around lip, eye, nose, and eyebrow, which coincides with the psychological studies [3]. It is also interesting that the patches selected for the expressions are closely related to a set of facial Action Units (AUs) [5], which can be used to describe the corresponding expression. For example, as shown in Fig. 4, the patches selected for recognizing the sadness expression are either located around the lip, which is closely related with AU 15 (Lip Corner Depressor), or around the eye corners and eyebrows, which are related to AU 4 (Brow Lowerer) and AU 1 (Inner Brow Raiser), respectively. The combination of AU 1, AU 4 and AU 15 describes the sadness expression [17]. Similar results can be found in other expressions.



(a) Ang (b) Dis (c) Fea (d) Hap (e) Sad (f) Sur

Figure 4. An analysis of the selected features for the six basic expressions in CK+ database. Red color means selection with the highest frequency, i.e., the feature was selected in all 8 runs; while blue color stands for relatively lower selection frequency, i.e., the feature was selected in more than 4 runs. Best viewed in color.

Another interesting discovery is that the number of selected patches decreases as BDBN learning continues. Starting from dozens patches selected in the first iteration, fewer and fewer patches are chosen. Finally, a small set of features (usually less than 7) was employed in the final strong classifier. Furthermore, the discriminative powers of the selected features were strengthened drastically. As shown in Fig. 5, an 80-dimensional vector is employed to store the individual recognition rates of all features

[5] We did not recognize the "contempt" and "neutral" for a fair comparison with the state-of-the-art methods evaluated on the original Cohn-Kanade database [11].

[6] Cohn-Kanade database [11] is an early version of CK+ and contains a subset of CK+ data (i.e., 320 image sequences with expression labels [23]).

(patches), where an "X" means the feature is not selected. We can find that most of the less expression-related features (e.g., patches around hair and neck) were deselected with learning going on; and individual recognition rates of the selected features (patches), which are shown as the numbers in the corresponding vector, increase as iteration goes.

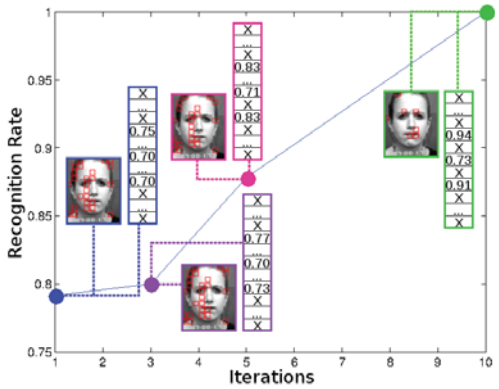


Figure 5. Recognition rate of the strong classifier increases with a decrease in the number of selected patches as iteration goes. More importantly, individual recognition rates for the selected features (patches) increase as well. An 80-dimensional vector is employed to store the individual recognition rates of all features (patches), where an "X" means the feature is not selected. Best viewed in color.

4.2.3 Computational Complexity

Our expression recognition system consists of an offline training phase and an online recognition phase. The offline training is performed in two steps: an initialization process for constructing 80 DBNs and a joint feature learning process via the BDBN with the BU-UFL and the BTD-SFS running alternatively. It took about 8 days to complete the overall training for 6 expressions in an 8-fold experimental setup on a 6-core 2.4GHZ PC using Matlab implementation. Note that BDBN training became more and more efficient as the learning continued because the number of selected patches kept decreasing until converge. In our experiments, less than 7 weak classifiers (selected patches) were employed in most of the final strong classifiers. For the online recognition, the average running time for each image is nearly 30ms $\times$ number of weak classifiers using Matlab implementation. In addition, the online recognition can be performed in real-time using a parallel computing strategy.

4.3. Experiments on the JAFFE Database

JAFFE database [18] consists of **213** images from 10 Japanese female subjects. Each subject has 3 or 4 examples of each of the six basic expressions and one sample of a neutral expression. The experimental results on the JAFFE database are used to demonstrate the cross-database generalization ability of the proposed method.

4.3.1 Cross-database Validation

To evaluate the generalization ability, we performed a cross-database validation, i.e., we trained the BDBN framework on the CK+ database and tested its performance on the JAFFE database. It is well received that the generalization across database is usually low. Shan et al [23] trained selected LBP features using SVMs on Cohn-Kanade database and tested the trained system on the JAFFE database, and obtained a classification rate about $\mathbf{41\%}$ for 7 expressions (6 basis expressions and neutral). From Table 2, we can find that the performance of BDBN is much higher than [23], which demonstrates that the features learned by BDBN capture the most critical expression-related information that can be generalized across different data sets.

Table 2. Cross-database validation, trained on CK+ database and tested on the JAFFE database, in terms of average classification rate for 7 expressions (6 basis expressions and neutral). In [23], LBP features were employed and fed into SVM with three different kernels, i.e., linear, polynomial, RBF, respectively.

Ada+SVM(Linear) [23]	Ada+SVM(Poly) [23]	Ada+SVM(RBF) [23]	**BDBN**
0.404	0.404	0.413	**0.680**

Table 3. Performance comparison on the JAFFE database in terms of average classification rate for 7 expressions (6 basis expressions and neutral). BDBN$_J$ was trained on images only from JAFFE; while BDBN$_{J+C}$ was trained on CK+ first and refined using the images in JAFFE.

SLLE [14]	SFRCS [12]	Ada+SVM(RBF) [23]	BDBN$_J$	**BDBN$_{J+C}$**
0.868	0.860	0.810	0.918	**0.930**

4.3.2 Performance Evaluation on the JAFFE Database

We also evaluated the BDBN framework trained and tested on the JAFFE database with a leave-one-subject-out training/testing strategy. We employed two settings: the first one, denoted as "BDBN with JAFFE Only" (BDBN$_J$), employed only the images in JAFFE for training, and the other one, denoted as "BDBN with JAFFE and CK+" (BDBN$_{J+C}$), was trained on the CK+ database first and then refined using the images in JAFFE. To make a fair comparison, we only compared with the-state-of-the-art methods employing the leave-one-subject-out strategy and recognizing 7 expressions. As shown in Table 3, the BDBN with both settings outperformed the other methods. Moreover, the BDBN$_{J+C}$ achieved the best performance, which implies that a learned BDBN can be effectively adapted to a new dataset with additional training data.

5. Conclusion and Future Work

In this work, we propose a novel BDBN framework to combine feature learning/strengthen, feature selection, and classifier construction in a unified framework. Specifically, features are fine-tuned jointly and are selected to form a strong classifier in a novel BTD-SFS process. Through this framework, highly complex features can be learned from facial images, and more importantly, the discriminative capabilities of selected features are strengthened iteratively according to their relative importance to the strong classifier. As demonstrated in the experiments, the BDBN learning framework outperformed all methods in comparison including the state-of-the-art techniques evaluated on two public

facial expression databases. There are several future directions to extend this framework. First, we will evaluate BDBN in more challenging scenarios, e.g., more spontaneous expressions with face pose variations. Second, this BDBN framework can be immediately employed in other classification problems, such as recognizing facial action units. Finally, we expect to extend the framework to handle video data, from which dynamic aspect of facial expressions can be captured and employed.

6. Acknowledgments

This work was supported by National Science Foundation under CAREER Award IIS-1149787.

References

[1] M. S. Bartlett, G. Littlewort, M. G. Frank, C. Lainscsek, I. Fasel, and J. R. Movellan. Recognizing facial expression: Machine learning and application to spontaneous behavior. In *CVPR*, volume 2, pages 568–573, 2005. 1, 2, 3, 6

[2] I. Bociu and I. Pitas. A new sparse image representation algorithm applied to facial expression recognition. In *MLSP*, pages 539–548, 2004. 1, 2

[3] J. F. Cohn and A. Zlochower. A computerized analysis of facial expression: Feasibility of automated discrimination. *American Psychological Society*, 1995. 2, 6

[4] M. Dahmane and J. Meunier. Emotion recognition using dynamic grid-based HoG features. In *FG*, March 2011. 1, 2

[5] P. Ekman, W. V. Friesen, and J. C. Hager. *Facial Action Coding System: the Manual*. Research Nexus, Div., Network Information Research Corp., Salt Lake City, UT, 2002. 6

[6] M. M. H, Z. Mu, V. K. L, M. S. Mohammad, and C. J. F. Facial action unit recognition with sparse representation. In *FG*, pages 336–342. IEEE, 2011. 1, 2

[7] G. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006. 4, 5

[8] G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. 3, 4, 5, 6

[9] Y. Hu, Z. Zeng, L. Yin, X. Wei, X. Zhou, and T. S. Huang. Multi-view facial expression recognition. In *FG*, pages 1–6, 2008. 1, 2

[10] S. Jain, C. Hu, and J. K. Aggarwal. Facial expression recognition with temporal modeling of shapes. In *ICCV Workshops*, pages 1642–1649, 2011. 1, 2

[11] T. Kanade, J. F. Cohn, and Y. Tian. Comprehensive database for facial expression analysis. In *FG*, pages 46–53, 2000. 2, 5, 6

[12] M. Kyperountas, A. Tefas, and I. Pitas. Salient feature and reliable classifier selection for facial expression classification. *Pattern Recognition*, 43(3):972–986, 2010. 7

[13] A. Lőrincz, L. A. Jeni, Z. Szabó, J. F. Cohn, and T. Kanade. Emotional expression classification using time-series kernels. In *CVPR Workshops*, pages 889–895. IEEE, 2013. 1, 2

[14] D. Liang, J. Yang, Z. Zheng, and Y. Chang. A facial expression recognition system based on supervised locally linear embedding. *Pattern Recognition Letters*, 26(15):2374–2389, 2005. 7

[15] Y. Lin, M. Song, D. Quynh, Y. He, and C. Chen. Sparse coding for flexible, robust 3d facial-expression synthesis. *Computer Graphics and Applications*, 32(2):76–88, 2012. 1, 2

[16] W. Liu, C. Song, and Y. Wang. Facial expression recognition based on discriminative dictionary learning. In *ICPR*, 2012. 1, 2

[17] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The extended cohn-kanade dataset (ck+): A complete expression dataset for action unit and emotion-specified expression. In *CVPR Workshops*, pages 94–101, 2010. 2, 5, 6

[18] M. J. Lyons, J. Budynek, and S. Akamatsu. Automatic classification of single facial images. *IEEE T-PAMI*, 21(12):1357–1362, 1999. 2, 5, 7

[19] M. Pantic, A. Pentland, A. Nijholt, and T. S. Huang. Human computing and machine understanding of human behavior: A survey. In T. S. Huang, A. Nijholt, M. Pantic, and A. Pentland, editors, *Artificial Intelligence for Human Computing*, LNAI. 2007. 2

[20] M. Ranzato, J. Susskind, V. Mnih, and G. Hinton. On deep generative models with applications to recognition. In *CVPR*, pages 2857–2864. IEEE, 2011. 1, 2, 3

[21] S. Rifai, Y. Bengio, A. Courville, P. Vincent, and M. Mirza. Disentangling factors of variation for facial expression recognition. In *ECCV*, pages 808–822. Springer, 2012. 1, 2, 3

[22] T. Senechal, V. Rapp, H. Salam, R. Seguier, K. Bailly, and L. Prevost. Combining AAM coefficients with LGBP histograms in the multi-kernel SVM framework to detect facial action units. In *FG Workshops*, pages 860 – 865, 2011. 1, 2

[23] C. Shan, S. Gong, and P. McOwan. Facial expression recognition based on Local Binary Patterns: A comprehensive study. *J. IVC*, 27(6):803–816, 2009. 6, 7

[24] J. M. Susskind, G. E. Hinton, J. R. Movellan, and A. K. Anderson. Generating facial expressions with deep belief nets. In V. Kordic, editor, *Affective Computing, Emotion Modelling, Synthesis and Recognition*, pages 421–440. 2008. 2

[25] Y. Tian, T. Kanade, and J. F. Cohn. Evaluation of Gabor-wavelet-based facial action unit recognition in image sequences of increasing complexity. In *FG*, pages 229–234, May 2002. 1, 2

[26] M. F. Valstar, M. Mehu, B. Jiang, M. Pantic, and K. Scherer. Meta-analyis of the first facial expression recognition challenge. *IEEE T-SMC-B*, 42(4):966–979, 2012. 1, 2

[27] J. Whitehill, M. S. Bartlett, G. Littlewort, I. Fasel, and J. R. Movellan. Towards practical smile detection. *IEEE T-PAMI*, 31(11):2106–2111, Nov. 2009. 1, 2, 3

[28] P. Yang, Q. Liu, and D. N. Metaxas. Boosting coded dynamic features for facial action units and facial expression recognition. In *CVPR*, pages 1–6, June 2007. 1, 2

[29] Z.-L. Ying, Z.-W. Wang, and M.-W. Huang. Facial expression recognition based on fusion of sparse representation. In D.-S. Huang, X. Zhang, C. Reyes García, and L. Zhang, editors, *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence*, LNCS, pages 457–464. 2010. 1, 2

[30] S. Zafeiriou and M. Petrou. Nonlinear non-negative component analysis algorithms. *IEEE T-IP*, 19(4):1050–1066, 2010. 1, 2

[31] Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *IEEE T-PAMI*, 31(1):39–58, Jan. 2009. 2

[32] Y. Zhang and Q. Ji. Active and dynamic information fusion for facial expression understanding from image sequences. *IEEE T-PAMI*, 27(5):699–714, May 2005. 1, 2

[33] Z. Zhang, M. Lyons, M. Schuster, and S. Akamatsu. Comparison between geometry-based and Gabor-wavelets-based facial expression recognition using multi-layer perceptron. In *FG*, pages 454–459, 1998. 1, 2

[34] G. Zhao and M. Pietiäinen. Dynamic texture recognition using local binary patterns with an application to facial expressions. *IEEE T-PAMI*, 29(6):915–928, June 2007. 1, 2

[35] R. Zhi, M. Flierl, Q. Ruan, and W. Kleijn. Graph-preserving sparse nonnegative matrix factorization with application to facial expression recognition. *IEEE T-SMC-B*, (99):1–15, 2010. 1, 2

[36] L. Zhong, Q. Liu, P. Yang, B. Liu, J. Huang, and D. Metaxas. Learning active facial patches for expression analysis. In *CVPR*, 2012. 1, 2, 3, 6