Joint Fine-Tuning in Deep Neural Networks for Facial Expression Recognition

Heechul Jung Sihaeng Lee Junho Yim Sunjeong Park Junmo Kim
School of Electrical Engineering
Korea Advanced Institute of Science and Technology
{heechul, haeng, junho.yim, sunny0414, junmo.kim}@kaist.ac.kr

Abstract

Temporal information has useful features for recognizing facial expressions. However, to manually design useful features requires a lot of effort. In this paper, to reduce this effort, a deep learning technique, which is regarded as a tool to automatically extract useful features from raw data, is adopted. Our deep network is based on two different models. The first deep network extracts temporal appearance features from image sequences, while the other deep network extracts temporal geometry features from temporal facial landmark points. These two models are combined using a new integration method in order to boost the performance of the facial expression recognition. Through several experiments, we show that the two models cooperate with each other. As a result, we achieve superior performance to other state-of-the-art methods in the CK+ and Oulu-CASIA databases. Furthermore, we show that our new integration method gives more accurate results than traditional methods, such as a weighted summation and a feature concatenation method.

1. Introduction

Recognizing an emotion from a facial image is a classic problem in the field of computer vision, and many studies have been conducted. It can be classified into two categories: image sequence-based and still image-based approaches. Image sequence-based approach has been used to increase the recognition performance by extracting useful temporal features from the image sequences, and the performance is usually better than a still image-based approach [15, 20, 12, 17, 8]. Both appearance and geometric features can be used for the spatio-temporal feature [1, 22].

Well-known deep learning algorithms, such as the deep neural networks (DNNs) and the convolutional neural networks (CNNs), have an ability to automatically extract useful representations from raw data (e.g., image data). However, there is a limit when applying them directly to facial expression recognition databases, such as CK+ [13], MMI [18], and Oulu-CASIA [23]. The major reason is that the amount of data is too small, so a deep network that has many parameters can easily fall into overfitting when training. (In general, data collection is expensive.) Furthermore, if the training data is high dimensional, the overfitting problem becomes more crucial.

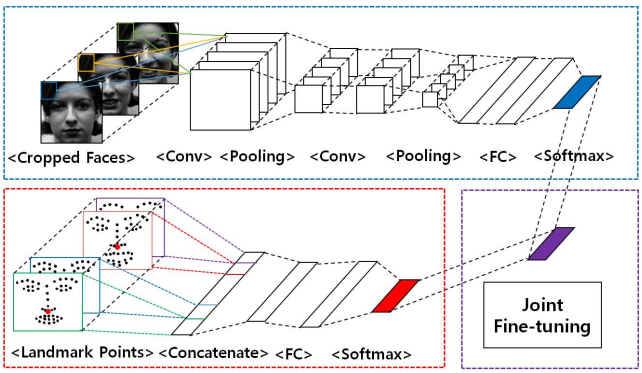


Figure 1. **Overall structure of our approach.** The blue and red boxes with dotted lines correspond to the two architectures of the deep networks. Our two deep networks receive an image sequence and facial landmark points as input, respectively. Conv and FC refer to the convolutional and fully connected layers. Finally, the outputs of these networks are integrated using a proposed joint fine-tuning method, which is represented in the purple box.

In this paper, we are interested in recognizing facial expressions using a limited amount of (typically a few hundreds of) image sequence data with a deep network. In order to overcome the problem of having a small amount of data, we construct two small deep networks that complement each other. One of the deep networks is trained using image sequences, while the other deep network learns the temporal trajectories of facial landmark points. In other words, the first network focuses more on appearance changes of facial expressions over time, while the second network is directly related to the motion of facial parts. Furthermore, we present a new integration method called joint fine-tuning, which performs better than simple weighted summation

1550-5499/15 $31.00 © 2015 IEEE
DOI 10.1109/ICCV.2015.341

method. Therefore, our main contributions in this paper can be summarized as follows:

- Two deep network models are presented in order **to extract useful temporal representations from two kinds of sequential data**: image sequences and the trajectories of landmark points.
- We observed that the two networks **automatically detect moving facial parts and action points**, respectively.
- We presented a **joint fine-tuning method** integrating these two networks with different characteristics, and **performance improvement** was achieved in terms of the recognition rates.

2. Related Work

2.1. Deep Learning-Based Method

Typically, a CNN uses a single image, but CNN can also be used for temporal recognition problems, such as action recognition [15]. In this 3D CNN method, the filters are shared along the time axis. Additionally, this method has been applied to facial expression recognition with deformable action part constraints, which is called 3D CNN-DAP [11]. The 3D CNN-DAP method is based on 3D CNN and uses the strong spatial structural constraints of the dynamic action parts. It could receive a performance boost from using the hybrid method, but it falls short of the performance of other state-of-the-art methods.

2.2. Hand-Crafted Feature-Based Method

Many studies in this field have been conducted. Traditional local features, such as HOG, SIFT, LBP, and BoW have been extended in order to be applicable to video, and these are called 3D HOG [9], 3D SIFT [16], LBP-TOP [24], and BoW [17], respectively. Additionally, there was an attempt to improve accuracy through temporal modeling of each facial shape (TMS) [6]. They used conditional random fields and shape-appearance features created manually.

Recently, spatio-temporal covariance descriptors with the Riemannian locality preserving projection approach were developed (Cov3D) [15], and an interval temporal Bayesian network (ITBN) for capturing complex spatio-temporal relations among muscles was proposed [20]. Recently, expressionlet-based spatio-temporal manifold representation was developed (STM-ExpLet) [12].

In addition to the ones mentioned above, perceptual color space was considered for facial expression [10]. Appearance and geometric feature-based approaches were also considered to be a solution for facial expression recognition, so several approches were developed [1, 22]. Recently, there is an approach that uses 3D shape model, and they improved facial expression recognition rate [8]. They used a ZFace algorithm [7] to estimate 3D landmark points, so the algorithm requires 3D information of face for training.

3. Our Approach

We utilize deep learning techniques in order to recognize facial expressions. Basically, two deep networks are combined: the deep temporal appearance network (DTAN) and the deep temporal geometry network (DTGN). The DTAN, which is based on a CNN, is used to extract the temporal appearance feature necessary for facial expression recognition. The DTGN, which is based on a fully connected DNN, catches geometrical information about the motion of the facial landmark points. Finally, these two models are integrated in order to increase the expression recognition performance. This network is called the deep temporal appearance-geometry network (DTAGN). The architecture of our deep network is shown in Figure 1.

3.1. Preprocessing

In general, the length of image sequences is variable, but the input dimension is usually fixed in a deep network. Consequently, the normalization along the time axis is required as input for the networks. We adopt the method in [26], which makes an image sequence into a fixed length. Then, the faces in the input image sequences are detected, cropped, and rescaled to 64×64. From these detected faces, facial landmark points are extracted using the algorithm called IntraFace [21]. This algorithm provides accurate facial landmark points consisting of 49 landmark points, including two eyes, a nose, a mouth, and two eyebrows.

3.2. Deep Temporal Appearance Network

In this paper, a CNN is used for capturing temporal changes of appearance. Conventional CNN uses still images as input, and 3D CNN was presented recently for dealing with image sequences. As mentioned in Section 2, the 3D CNN method shares the 3D filters along the time axis [15]. However, we use the n-image sequences without weight sharing along the time axis. This means that each filter plays a different role depending on the time. The activation value of the first layer is defined as follows:

$$f_{x,y,i} = \sigma\Big(\sum_{t=1}^{T_a}\sum_{r=0}^{R}\sum_{s=0}^{S} I^{(t)}_{x+r,y+s} \cdot w^{(t)}_{r,s,i} + b_i\Big), \quad (1)$$

where $f_{x,y,i}$ is the activation value of position (x, y) of the i-th feature map. R and S are the number of rows and columns of the filter, respectively. T_a is the total frame number of the input grayscale image sequences. $I^{(t)}_{x+r,y+s}$ means that the value at the position $(x+r, y+s)$ of the input frame at time t. $w^{(t)}_{r,s,i}$ is the i-th filter coefficient at (r, s) for the t-th frame, and b_i is the bias for the i-th filter. $\sigma(\cdot)$ is an

activation function, which is usually a non-linear function. Additionally, we utilize a ReLU, $\sigma(x) = max(0, x)$ as an activation function, where x is an input value [3].

The other layers are not different from the conventional CNN as follows: the output of the convolutional layer is rescaled to half-size in a pooling layer for efficient calculation. Using these activation values, a convolution operation and pooling are performed one more time. Finally, these output values are passed through the two fully connected layers and then classified using softmax. For training our network, the stochastic gradient descent method is used for optimization, and dropout [5] and weight decay methods are utilized for regularization.

We designed our network with a moderate depth and a moderate number of parameters to avoid overfitting, since the size of the facial expression recognition database is too small— there are only 205 sequences in the MMI database. Additionally, the first layer turns out to detect the temporal difference of the appearance in input image sequences as discussed in Section 4.

3.3. Deep Temporal Geometry Network

DTGN receives the trajectories of facial landmark points as input. These trajectories can be considered as one-dimensional signals and defined as follows:

$$X^{(t)} = \begin{bmatrix} x_1^{(t)} & y_1^{(t)} & x_2^{(t)} & y_2^{(t)} & \cdots & x_n^{(t)} & y_n^{(t)} \end{bmatrix}^\top, \tag{2}$$

where n is the total number of landmark points at frame t, and $X^{(t)}$ is a $2n$ dimensional vector at t. $x_k^{(t)}$ and $y_k^{(t)}$ are coordinates of the k-th facial landmark points at frame t.

These xy-coordinates are inappropriate for direct use as an input to the deep network, because they are not normalized. For the normalization of the xy-coordinates, we first subtract the xy-coordinates of the nose position (the position of the red point among the facial landmark points in the red box with the dotted line in Figure 1) from the xy-coordinates of each point. Then, each coordinate is divided by each standard deviation of xy-coordinates in each frame as follows:

$$\bar{x_i}^{(t)} = \frac{x_i^{(t)} - x_o^{(t)}}{\sigma_x^{(t)}}, \tag{3}$$

where $x_i^{(t)}$ is x-coordinate of the i-th facial landmark point at frame t, $x_o^{(t)}$ is x-coordinate of the nose landmark coordinate at frame t. $\sigma_x^{(t)}$ is standard deviation of x-coordinates at frame t. This process is also applied to the $y_i^{(t)}$. Finally, these normalized points are concatenated along the time, and these points are used for the input to the DTGN.

$$\bar{X} = \begin{bmatrix} \bar{x}_1^{(1)} & \bar{y}_1^{(1)} & \cdots & \bar{x}_n^{(T_g)} & \bar{y}_n^{(T_g)} \end{bmatrix}^\top, \tag{4}$$

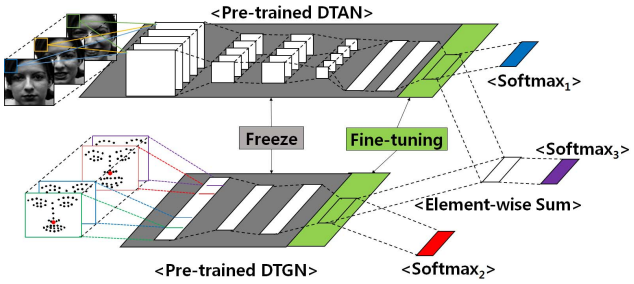


Figure 2. **Joint fine-tuning method.** The green box denotes linear fully connected network which has logit values. The logit values are used as the input to the softmax activation. To integrate two networks, we freeze the weight values in gray boxes of two trained networks, and retrain the top layer in green boxes. In the training step, we use three softmax functions for calculating three loss functions, and we only use Softmax_3 for prediction.

where $\bar{X}$ is a $2nT_g$ dimensional input vector, and $\bar{x}_k^{(T_g)}$ and $\bar{y}_k^{(T_g)}$ are coordinates of k-th normalized landmark points at frame T_g.

The figure in the red box with a dotted line in Figure 1 illustrates the architecture of our DTGN model. Our network receives the concatenated landmark points $\bar{X}$ as input. Basically, we utilize two hidden layers, and the top layer is a softmax layer. Similar to the DTAN, this network is also trained by using the stochastic gradient descent method. The activation function for each hidden layer is ReLU. Furthermore, for regularization of the network, dropout [5] and weight decay are used.

3.4. Data Augmentation

In order to better classify unseen data, a number of training data covering various situations are required. However, facial expression databases, such as CK+, Oulu-CASIA, and MMI, provide only hundreds of sequences. This makes a deep network easily overfit, because a typical deep network has many parameters. To overcome this problem, various data augmentation techniques are required.

First, whole image sequences are horizontally flipped. Then, each image is rotated by each angle in $\{-15°, -10°, -5°, 5°, 10°, 15°\}$. This makes the model robust against the slight rotational changes of the input images. Finally, we obtain 14 times more data: original images (1), flipped images (1), rotated images with six angles, and their flipped versions (12).

Similar to the augmentation of image sequences, the normalized facial landmark points are also horizontally flipped. Then, Gaussian noise is added to the raw landmark points.

$$\tilde{x}_i^{(t)} = \bar{x}_i^{(t)} + z_i^{(t)}, \tag{5}$$

where $z_i^{(t)} \sim N(0, \sigma_i^2)$ is additive noise with noise level σ_i for the x-coordinate of the i-th landmark points at frame t.

We set the value of σ_i to 0.01. Additionally, we contaminated y-coordinate with noise in the same way. The network learns to be robust against slight pose changes using this method. To prepare for rotational changes, we construct rotated data as follows:

$$\begin{bmatrix} \tilde{x}_i^{(t)} & \tilde{y}_i^{(t)} \end{bmatrix}^\top = R^{(t)} \begin{bmatrix} \bar{x}_i^{(t)} & \bar{y}_i^{(t)} \end{bmatrix}^\top, \quad (6)$$

for $i = 1, \ldots, n$ where $\tilde{x}_i^{(t)}$ and $\tilde{y}_i^{(t)}$ are i-th rotated xy-coordinates at time t, and $R^{(t)}$ is a 2×2 rotation matrix for the xy-coordinates at time t, which has an angle $\theta^{(t)}$. The value of $\theta^{(t)}$ is drawn from a uniform distribution where $\theta^{(t)} \sim \text{Unif}[\beta, \gamma]$. We set the values of β and γ to $-\pi/10$ and $\pi/10$, respectively.

We performed the first data augmentation methods in equation 5 three times, and the second data augmentation in equation 6 was also conducted three times. Consequently, we obtained six times more facial landmark points. As a result, we augmented the training data fourteen times: original coordinates (1), flipped coordinates (1), and six augmented coordinates, and their flipped versions (12).

3.5. Model Integration

3.5.1 Weighted Summation

The outputs from the top layers of the two networks were integrated using equation 7.

$$o_i = \alpha p_i + (1 - \alpha) q_i, \ 0 \leq \alpha \leq 1, \quad (7)$$

for $i = 1, \ldots, c$ where c is the total number of emotion class, p_i, q_i are outputs of DTAN and DTGN, and o_i is the final score. Finally, the index with the maximum value is the final prediction. The parameter α usually depends on the performance of each network. For all the experiments in this paper, we set the value of α to 0.5 that is the optimal value as shown in Figure 11.

3.5.2 Joint Fine-Tuning Method

The above method is simple to use, but it may not use the most of the ability of the two models. Consequently, we propose an alternative integration method for the two networks using a joint fine-tuning method, which achieves better results than the above method.

First, the two trained networks are reused, as shown in Figure 2. Next, we retrain the linear fully connected network, which is located below softmax activation function, with the loss function L_{DTAGN} of DTAGN defined as follows:

$$L_{DTAGN} = \lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3, \quad (8)$$

where L_1, L_2, and L_3 are loss functions computed by DTAN, DTGN, and both, respectively. The λ_1, λ_2, and λ_3 are tuning parameters. Usually, the parameters λ_1 and λ_2 are the same, and λ_3 has a smaller value than the value of two tuning parameters. For all the experiments perfomed in this paper, we set λ_1, λ_2, and λ_3 to 1, 1, and 0.1, respectively. The parameters were intuitively chosen. Each loss function is a cross entropy loss function, which is defined as follows:

$$L_i = -\sum_{j=1}^{c} y_j \log(\tilde{y}_{i,j}), \quad (9)$$

where y_j is the j-th value of the ground truth label, and $\tilde{y}_{i,j}$ is the j-th output value of softmax of network i. (For convenience, we call DTAN, DTGN, and the integrated network by network $1, 2$, and 3, respectively.) The $\tilde{y}_{3,j}$ is defined using logit values of network 1 and 2 as follows:

$$\tilde{y}_{3,j} = \sigma_s(l_{1,j} + l_{2,j}), \quad (10)$$

where $l_{1,j}$ and $l_{2,j}$ are j-th logit values of network 1 and 2, respectively. $\sigma_s(\cdot)$ is a softmax activation function.

Finally, the final decision $\tilde{o}$ is obtained using the output of softmax of network 3 as follows:

$$\tilde{o} = \arg\max_j \tilde{y}_{3,j}, \quad (11)$$

As a result, we utilized three loss functions in the training step, and use only integrated result for prediction. When using our joint fine-tuning method, we use the same training dataset used in training of each network. Also, the dropout method is used for reducing over-fitting.

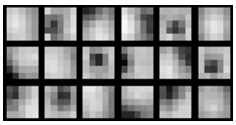


Figure 3. **Filters learned by a single frame-based CNN.** The input image size was 64 × 64, and the filter size was 5 × 5. 18 filters were selected for visualization from 64 learned filters in the first convolutional layer. The black and white colors represent the negative and positive values, respectively. There were several directional edge and blob detection filters.

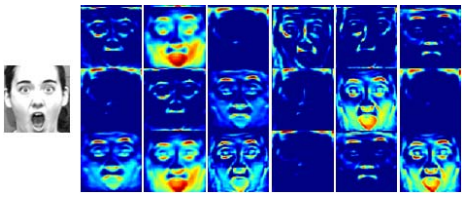


Figure 4. **Feature maps corresponding to Figure 3.** The left image represents the input image, and the right image shows the feature maps extracted by each filter in Figure 3. The emotion label for the input image was surprise. The blue and red values represent the low and high response values, respectively. The edges of the input image are detected in most of the filter.

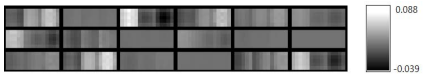


Figure 5. **Filters learned by DTAN.** The number of input frames was three in this figure, so there are three filters corresponding to each frame. The three filters in each bold black box generate one feature map. As with Figure 3, 18 filters were selected from 64 learned filters. In this figure, we can see that our network detects differences between frames.

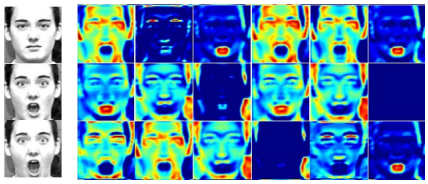


Figure 6. **Feature maps corresponding to Figure 5.** The gray images on the left side form the image sequence used as input, and the images on the right side are the feature maps corresponding to each filter in Figure 5. Blue and red represent the low and high response values. The emotion label for the input image sequence was surprise. We observed that our network responded to moving parts for expressing emotion.

4. What Will Deep Networks Learn?

4.1. Visualization of DTAN

To find out what our DTAN has learned, the learned filters were visualized. Figure 3 demonstrates the filters learned by a single frame-based CNN in the first convolutional layers using the CK+ database. The filters were similar to the edge or blob detectors. Corresponding responses to each filter are provided in Figure 4. The edge components with several directions were detected by these filters.

In our DTAN, which is a multiple frame-based CNN, the learned filters are shown in Figure 5. Unlike the filters of a single frame-based CNN, the filters were not edge or blob detectors. To exaggerate a little, these were just combinations of black, gray, and white filters. Figure 6 shows the meaning of these filters. High response values were usually shown in parts with big differences between input frames. In other words, we can see that the first convolutional layer of our DTAN detects facial movements arising from the expression of emotion.

4.2. Visualization of DTGN

The left side of Figure 7 (a) shows the significant facial landmark points for facial expression recognition. These positions were automatically identified by DTGN. The extracted positions were very similar to those of emotional facial action coding system (EFACS) [2] in Figure 7 (b). To explain it further, the two extracted points on the nose become wider when people make a happy expression because both cheeks are pulled up.

In order to figure out the characteristics of the features extracted from the top layer, we also visualized the feature vectors using t-SNE, which is a useful tool for visualization of high dimensional data [19]. The input data were spread randomly in Figure 7 (c), but the features extracted from the second hidden layer were well separated according to their label, as shown in Figure 7 (d).

	An	Co	Di	Fe	Ha	Sa	Su	All
CK+	45	18	59	25	69	28	83	327
Oulu	80	-	80	80	80	80	80	480
MMI	32	-	31	28	42	32	40	205

Table 1. **The number of image sequences for each emotion**: anger (An), contempt (Co), disgust (Di), fear (Fe), happiness (Ha), sadness (Sa), and surprise (Su).

5. Experiments

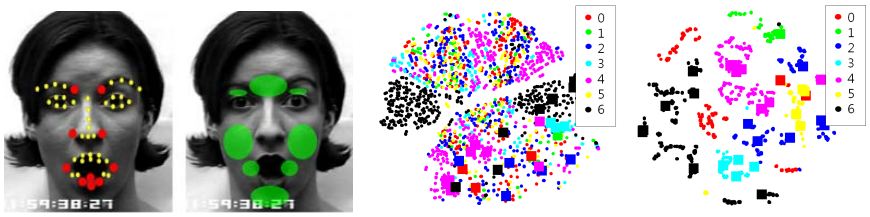
In this section, we compare our approach with other state-of-the-art algorithms in facial expression recognition, such as manifold-based sparse representation (MSR) [14], AdaLBP [23], Atlases [4], and common and specific active patches (CSPL) [25]. We excluded person dependent algorithms or algorithms that utilize 3D geometry information in the experiments. For assessing the performance of our method, we used three databases: the CK+, Oulu-CASIA, and MMI databases. The number of image sequences in each database is listed according to each emotion in Table 1.

5.1. Network Architecture

The architecture of DTGN for the CK+ is D1176-FC100-FC600-S7. D1176 is a 1176 dimensional input vector, and FC100 refers to a fully connected layer with 100 nodes. Also, S7 is the softmax layer with seven outputs. Our DTAN model for the CK+ is I64-C(5,64)-L5-P2-C(5,64)-L3-P2-FC500-FC500-S7, where I64 means 64×64 input image sequences, and C(5,64) is a convolutional layer with 64 filters of 5×5. L5 is a local contrast normalization layer with a window size of 5×5. P2 means a 2×2 max pooling layer. The stride of each layer was 1 with the exception of the pooling layer. The value of the stride for each pooling layer was set to 2. The DTGN and DTAN models

Method	Accuracy
HOG 3D [9]	91.44
MSR [14]	91.4
TMS [6]	91.89
Cov3D [15]	92.3
STM-ExpLet [12]	94.19
3DCNN [11]	85.9
3DCNN-DAP [11]	92.4
DTAN	91.44
DTGN	92.35
DTAGN(Weighted Sum)	**96.94**
DTAGN(Joint)	**97.25**

Table 2. **Overall accuracy in the CK+ database.** The red and blue colors represent the first and second most accurate, respectively.



(a) (b) (c) (d)

Figure 7. **Visualization of representation extracted by DTGN.** (a) (b) show extracted feature points and emotional action parts [2], respectively. The important top-10 positions are detected by our network (red points in the left figure). In order to visualize these ten points, we calculated the average of the absolute values of weights in the first layer connected to each landmark point. Then, these values were sorted in descending order, and the top 10 points with highest value were selected. The action parts defined by EFACS [2] are shown in the right figure (green colored area). (c) Visualization of original input data in CK+ database, using t-SNE [19]. The number of data was 4149: (327-33)×14 augmented training data and 33 test data. The small dots and large squares represent training and test data, respectively. The numbers in the legend correspond to each label of the CK+ database: 0-anger, 1-contempt, 2-disgust, 3-fear, 4-happiness, 5-sadness, and 6-surprise. (d) Visualization of the outputs in the second hidden layer. The data points were automatically grouped by DTGN.

for Oulu-CASIA were the same as the models for the CK+ except the number of nodes in the top layer, because there are six labels in the Oulu-CASIA.

For the MMI, we used the DTGN model of D1176-FC100-FC200-FC6. Our DTAN model was designed as I64-C(5,32)-P3-C(3,32)-FC30-S6. Unlike the other two databases, the number of subjects and image sequences are very small. Consequently, we decreased the total number of parameters significantly.

5.2. CK+

Description of the database. CK+ is a representative database for facial expression recognition. This database is composed of 327 image sequences with seven emotion labels: anger, contempt, disgust, fear, happiness, sadness, and surprise. There are 118 subjects, and these subjects are divided into ten groups by ID in ascending order. Nine subsets were used for training our networks, and the remaining subset was used for validation. This process is the same as the 10-fold cross validation protocol in [12]. In this database, each sequence starts with a neutral emotion and ends with a peak of the emotion.

Results. The total accuracy of 10-fold cross validation is shown in Table 2. The performances of DTAN and DTGN are lower than other algorithms, but the performance of the integrated network is better than other state-of-the-art algorithms. The two networks were complementary, and this is shown in Figure 8. The DTAN had a good performance with respect to contempt, whereas it had lower accuracy with fear. On the other hand, the geometry-based model was strong with fear. Table 3 shows the confusion matrix for CK+. Our algorithm performed well in recognizing anger, disgust, happiness, and surprise. For the other emotions, our method also performed reasonably well.

	An	Co	Di	Fe	Ha	Sa	Su
An	**100**	0	0	0	0	0	0
Co	0	**94.44**	0	0	0	5.56	0
Di	0	0	**100**	0	0	0	0
Fe	0	0	0	**84**	8	0	8
Ha	0	0	0	0	**100**	0	0
Sa	10.71	0	0	0	0	**89.29**	0
Su	0	1.2	0	0	0	0	**98.8**

Table 3. **Confusion matrix of the joint fine-tuning method for the CK+ database.** The labels in the leftmost column and on the top represent the ground truth and prediction results, respectively.

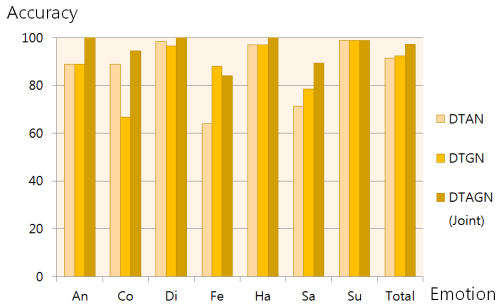


Figure 8. **Comparison of accuracy in the CK+** according to each emotion among three networks.

5.3. Oulu-CASIA

Description of the database. For further experiments, we used Oulu-CASIA, which includes 480 image sequences taken under normal illumination conditions. Each image sequence has one of six emotion labels: anger, disgust, fear, happiness, sadness, or surprise. There are 80 subjects, and 10-fold cross validation was performed in the same way as in the case of CK+. Similar to the CK+ database, each sequence begins with a neutral facial expression and ends with the facial expression of each emotion.

Results. The accuracy of our algorithm was superior to the other state-of-the-art algorithms, as shown in Table 4. The best performance from among the existing methods was 75.52%, which was achieved by Atlases, and this record had not been broken for three years. However, we have signif-

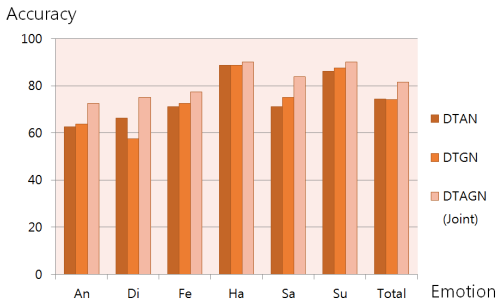


Figure 9. **Comparison of accuracy in the Oulu-CASIA** according to each emotion among three networks.

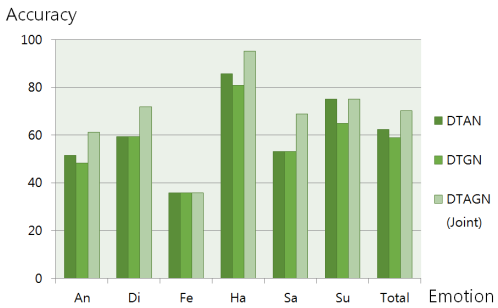


Figure 10. **Comparison of accuracy in the MMI** according to each emotion among three networks.

icantly improved the accuracy by about 6% using our integrated two deep networks. In Figure 9, the performance of two networks and the combined model is compared. Similar to the case of CK+, we can see that the two networks are complementary to each other. In particular, the performance of the DTGN in the case of disgust was lower than the DTAN, but the combined model produced good results. Table 5 shows the confusion matrix for our algorithm. The performance in the cases of happiness, sadness, and surprise was good, but the performance for anger, disgust, and fear was relatively poor. In particular, anger and disgust were confused in our algorithm.

5.4. MMI

Description of the database. MMI consists of 205 image sequences with frontal faces and includes only 30 subjects. Similar to the Oulu-CASIA database, there are six kinds of emotion labels. This database was also divided into 10 groups for person independent 10-fold cross validatiaon. This database is different from the other databases; each sequence begins with a neutral facial expression, and has the facial expression of each emotion in the middle of the sequence. This ends with the neutral facial expression. The location of the peak frame is not provided as a prior information.

Results. This dataset is especially difficult for a deep learning algorithm to learn from, because there are too small number of data and subjects. The previous top record achieved by a deep learning technique was only 63.4% using 3D CNN-DAP. However, we improved the recognition rate to 70.24% as shown in Table 6. Finally, our algorithm is much better than 3D SIFT, which was the second best algorithm. In particular, our joint fine-tuning method achieved a significantly improved recognition rate compared with a

Method	Accuracy
3D SIFT [16]	55.83
LBP-TOP [24]	68.13
HOG 3D [9]	70.63
AdaLBP [23]	73.54
Atlases [4]	75.52
STM-ExpLet [12]	74.59
DTAN	74.38
DTGN	74.17
DTAGN(Weighted Sum)	**80.62**
DTAGN(Joint)	**81.46**

Table 4. **Overall accuracy in the Oulu-CASIA database.** The red and blue colors represent the first and second most accurate, respectively.

	An	Di	Fe	Ha	Sa	Su
An	**72.5**	16.25	1.25	1.25	8.75	0
Di	21.25	**75**	3.75	0	0	0
Fe	2.5	1.25	**77.5**	6.25	2.5	10
Ha	0	0	7.5	**90**	2.5	0
Sa	13.75	0	2.5	0	**83.75**	0
Su	0	0	10	0	0	**90**

Table 5. **Confusion matrix of the joint fine-tuning method for the Oulu-CASIA database.** The labels in the leftmost column and on the top represent the ground truth and prediction results, respectively.

Method	Accuracy
HOG 3D [9]	60.89
3D SIFT [16]	64.39
ITBN [20]	59.7
CSPL [25]	(73.53)
STM-ExpLet [12]	**75.12**
3DCNN [11]	53.2
3DCNN-DAP [11]	63.4
DTAN	62.45
DTGN	59.02
DTAGN (Weighted Sum)	65.85
DTAGN (Joint)	**70.24**

Table 6. **Overall accuracy in the MMI database.** The red and blue colors represent the first and second most accurate, respectively. The CSPL used additional ground truth information, so it was excluded from the ranking.

	An	Di	Fe	Ha	Sa	Su
An	**61.29**	25.8	0	0	12.9	0
Di	15.62	**71.88**	0	9.37	0	3.13
Fe	10.71	0	**35.71**	10.71	14.29	28.57
He	0	0	4.76	**95.24**	0	0
Sa	9.38	3.13	15.62	0	**68.8**	3.12
Su	2.5	0	20	2.5	0	**75**

Table 7. **Confusion matrix of the joint fine-tuning method for the MMI Database.** The labels in the leftmost column and on the top represent the ground truth and prediction results, respectively.

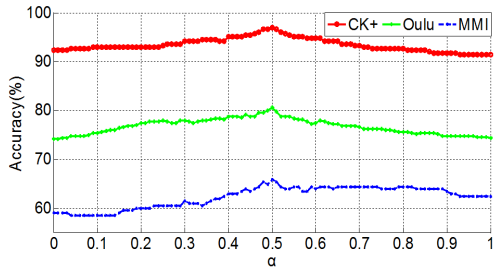


Figure 11. **Performance of DTAGN using the weighted summation method with respect to α.** We changed the value of α from 0 to 1 with interval of 0.01.

Figure 12. **All failure cases with fear in the MMI database.** Our deep network predicted fear to surprise (green box), anger (red box), sadness (orange box), and happiness (blue box).

weighted summation method.

We compared two networks and the combined model in Figure 10. The two networks were complementary to each other for most of the emotions. However, with fear, our algorithm was not successful enough. This is also shown in the confusion matrix in Table 7. We observed that the accuracy for fear was much lower than other emotions. In particular, most of the fear emotions were confused with surprise. To examine this phenomenon, we checked all the failure cases, as shown in Figure 12. The results indicated that a variety of facial expressions are labeled as fear, even though many cases were similar to surprise or sadness. To successfully recognize these various expressions, various kinds of training data are additionally required. However, we had only 27 subjects for training data. (Three subjects were used for validation.) Unfortunately, performance of deep learning techniques highly depends on the quality of training data, so our accuracy with fear was not good enough.

6. Discussion on the Joint Fine-Tuning Method

In this section, we discuss the joint fine-tuning method. First we evaluated the effectiveness of the three loss functions of our joint fine-tuning method using each database. As a result, the accuracy using the three loss functions was better than using L_3 only, as shown in Table 8. Also, we compared our algorithm with a concatenation of high level features, which is one natural way for integrating two deep networks. To evaluate the concatenation method, we concatenated the activation values of the top hidden layers in the two gray areas in Figure 2. Then the concatenated activation values are used for inputs to a fully connected network with a softmax activation, and a dropout was used for regularizing the network. Table 9 shows the experimental results using each database. The performance of the concatenation method was worse than either one of our two networks.

# Of Loss Functions	Baseline	One (L_3 only)	Three
Accuracy (CK+)	96.94	96.64	**97.25**
Accuracy (Oulu)	80.62	81.04	**81.46**
Accuracy (MMI)	65.85	69.76	**70.24**

Table 8. **Comparison between one and three loss function methods.** Baseline denotes the weighted summation method presented in this paper.

	Concatenation	Joint Fine-tuning
Accuracy (CK+)	94.5	**97.25**
Accuracy (Oulu)	75.63	**81.46**
Accuracy (MMI)	67.8	**70.24**

Table 9. **Comparison between the concatenation method and proposed joint fine-tuning method.** Our joint fine-tuning method showed about 3∼6% improvement in terms of the recognition rates.

As mentioned in Section 3.5.2, we used the same dataset for fine-tuning together with dropout, which reduces overfitting. Of course, without dropout, there will be little to fine-tune with the same dataset, as the error for the same training dataset would be already almost zero before fine-tuning starts. Interestingly, thanks to dropout, there is something to fine-tune with the same dataset as the dropout by randomly inducing errors achieves an effect of providing different training data to the layers to fine-tune. Further, as dropout creates an ensemble of many networks and each network in the ensemble experiences a different random subset of whole dataset, each member of ensemble with a particular dropout pattern experiences different subsets of data for the first training and the following fine-tuning.

7. Conclusion

We presented two deep network models that collaborate with each other. The first network was DTAN, which was based on appearances of multiple frames, while the second network was DTGN, which extracted useful temporal geometric features from raw facial landmark points. We showed that the filters learned by the DTAN in the first layer have the ability to obtain the difference between the input frames. Furthermore, the important landmark points extracted by DTGN were also shown. We achieved best recognition rates using the integrated deep network on the CK+ and Oulu-CASIA databases. Furthermore, we showed that our joint fine-tuning method is superior to other integration methods, such as a weighted summation and a feature concatenation method.

Acknowledgements This work was partially supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government (MSIP) NRF-2010-0028680 and NRF-2014R1A2A2A01003140.

References

[1] D. Ghimire and J. Lee. Geometric feature-based facial expression recognition in image sequences using multi-class adaboost and support vector machines. *Sensors*, 13(6):7714–7734, 2013. 1, 2

[2] J. Girard and J. Cohn. Ground truth facs action unit coding on the group formation task. In *Tech. rep., University of Pittsburgh*, 2013. 5, 6

[3] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP Volume*, volume 15, pages 315–323, 2011. 3

[4] Y. Guo, G. Zhao, and M. Pietikäinen. Dynamic facial expression recognition using longitudinal facial expression atlases. In *ECCV, 2012*, pages 631–644. Springer, 2012. 5, 7

[5] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012. 3

[6] S. Jain, C. Hu, and J. K. Aggarwal. Facial expression recognition with temporal modeling of shapes. In *ICCV Workshops, 2011*, pages 1642–1649. IEEE, 2011. 2, 5

[7] L. A. Jeni, J. F. Cohn, and T. Kanade. Dense 3d face alignment from 2d videos in real-time. 2

[8] L. A. Jeni, A. Lőrincz, Z. Szabó, J. F. Cohn, and T. Kanade. Spatio-temporal event classification using time-series kernel based structured sparsity. In *Computer Vision–ECCV 2014*, pages 135–150. Springer, 2014. 1, 2

[9] A. Klaser and M. Marszalek. A spatio-temporal descriptor based on 3d-gradients. 2008. 2, 5, 7

[10] S. M. Lajevardi and H. R. Wu. Facial expression recognition in perceptual color space. *Image Processing, IEEE Transactions on*, 21(8):3721–3733, 2012. 2

[11] M. Liu, S. Li, S. Shan, R. Wang, and X. Chen. Deeply learning deformable facial action parts model for dynamic expression analysis. In *ACCV, 2014*, pages 1749–1756. IEEE, 2014. 2, 5, 7

[12] M. Liu, S. Shan, R. Wang, and X. Chen. Learning expressionlets on spatio-temporal manifold for dynamic facial expression recognition. In *CVPR, 2014 IEEE Conference on*, pages 1749–1756. IEEE, 2014. 1, 2, 5, 6, 7

[13] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *CVPRW, 2010 IEEE Computer Society Conference on*, pages 94–101. IEEE, 2010. 1

[14] R. Ptucha, G. Tsagkatakis, and A. Savakis. Manifold based sparse representation for robust expression recognition without neutral subtraction. In *ICCV Workshops, 2011*, pages 2136–2143. IEEE, 2011. 5

[15] A. Sanin, C. Sanderson, M. T. Harandi, and B. C. Lovell. Spatio-temporal covariance descriptors for action and gesture recognition. In *WACV, 2013*, pages 103–110. IEEE, 2013. 1, 2, 5

[16] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th international conference on Multimedia*, pages 357–360. ACM, 2007. 2, 7

[17] K. Sikka, T. Wu, J. Susskind, and M. Bartlett. Exploring bag of words architectures in the facial expression domain. In *Computer Vision–ECCV 2012. Workshops and Demonstrations*, pages 250–259. Springer, 2012. 1, 2

[18] M. Valstar and M. Pantic. Induced disgust, happiness and surprise: an addition to the mmi facial expression database. In *Proc. Intl Conf. Language Resources and Evaluation, Workshop on EMOTION*, pages 65–70, 2010. 1

[19] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008. 5, 6

[20] Z. Wang, S. Wang, and Q. Ji. Capturing complex spatio-temporal relations among facial muscles for facial expression recognition. In *CVPR, 2013 IEEE Conference on*, pages 3422–3429. IEEE, 2013. 1, 2, 7

[21] X. Xiong and F. De la Torre. Supervised descent method and its applications to face alignment. In *CVPR, 2013 IEEE Conference on*, pages 532–539. IEEE, 2013. 2

[22] A. A. Youssif and W. A. Asker. Automatic facial expression recognition system based on geometric and appearance features. *Computer and Information Science*, 4(2):p115, 2011. 1, 2

[23] G. Zhao, X. Huang, M. Taini, S. Z. Li, and M. PietikäInen. Facial expression recognition from near-infrared videos. *Image and Vision Computing*, 29(9):607–619, 2011. 1, 5, 7

[24] G. Zhao and M. Pietikainen. Dynamic texture recognition using local binary patterns with an application to facial expressions. *PAMI*, 29(6):915–928, 2007. 2, 7

[25] L. Zhong, Q. Liu, P. Yang, B. Liu, J. Huang, and D. N. Metaxas. Learning active facial patches for expression analysis. In *CVPR, 2012 IEEE Conference on*, pages 2562–2569. IEEE, 2012. 5, 7

[26] Z. Zhou, G. Zhao, and M. Pietikainen. Towards a practical lipreading system. In *CVPR, 2011 IEEE Conference on*, pages 137–144. IEEE, 2011. 2