IPSJ Transactions on Computer Vision and Applications Vol.7 104–120 (Aug. 2015)

[DOI: 10.2197/ipsjtcva.7.104]

Regular Paper

Facial Expression Recognition and Analysis: A Comparison Study of Feature Descriptors

Chun Fui Liew[1,a)] Takehisa Yairi[1,b)]

Received: December 11, 2014, Accepted: May 15, 2015, Released: August 31, 2015

Abstract: Facial expression recognition (FER) is a crucial technology and a challenging task for human–computer interaction. Previous methods have been using different feature descriptors for FER and there is a lack of comparison study. In this paper, we aim to identify the best features descriptor for FER by empirically evaluating *five* feature descriptors, namely Gabor, Haar, Local Binary Pattern (LBP), Histogram of Oriented Gradients (HOG), and Binary Robust Independent Elementary Features (BRIEF) descriptors. We examine each feature descriptor by considering *six* classification methods, such as k-Nearest Neighbors (k-NN), Linear Discriminant Analysis (LDA), Support Vector Machine (SVM), and Adaptive Boosting (AdaBoost) with *four* unique facial expression datasets. In addition to test accuracies, we present confusion matrices of FER. We also analyze the effect of combined features and image resolutions on FER performance. Our study indicates that HOG descriptor works the best for FER when image resolution of a detected face is higher than 48×48 pixels.

Keywords: Facial expression recognition, Gabor filter, Haar filter, Local Binary Patterns, Histogram of Oriented Gradients, Binary Robust Independent Elementary Features

1. Introduction

Facial expression recognition (FER) has a great potential for improving our life quality. For instance, FER system is useful for medical applications, such as aiding patients with facial paralysis disease during rehabilitation treatment. FER system could also be used to analyze audience's facial expression for satisfaction survey. A robotic teacher could offer better learning experience by having a better understanding of students' feeling.

Designing a FER system is challenging due to the huge variability of face appearance, head pose, light condition, and partial occlusions due to hairs, sunglasses and masks. Over the years, researchers have proposed various techniques for robots/computers to recognize human facial expression. Previous works typically focus on proposing new feature descriptors and new classification methods for FER. In contrast, in this paper, we aim to identify the best feature descriptors by performing an extensive comparison study. We empirically evaluate *five* popular feature descriptors (Section 4), namely Gabor [1], Haar [2], Local Binary Pattern (LBP) [3], Histogram of Oriented Gradients (HOG) [4], and Binary Robust Independent Elementary Features (BRIEF) [5] descriptors. We then examine each feature descriptor by considering *six* classification methods (Section 5), such as k-Nearest Neighbors (k-NN), Linear Discriminant Analysis (LDA), Support Vector Machine (SVM), and Adaptive Boosting (AdaBoost) with *four* unique facial expression datasets. For clarity, we summarized the facial expression datasets, feature descriptors, and classification methods considered in our experiments in **Table 1**. In addition to test accuracies (Section 6), we present confusion matrices of FER (Section 8). After that, we analyze the effect of combined features (Section 9) and image resolutions (Section 10) on FER performance. We also generalized our experiments to other datasets (Section 11), analyzed the computational efficiency of each feature descriptors (Section 12), and visualized the feature descriptors selected by AdaBoost classifier (Section 13).

Table 1 Facial expression datasets, feature descriptors, and classification methods considered in our experiments.

Datasets	Descriptors	Classifiers
1. CK+	1. Gabor	1. k-NN
2. MUG	2. Haar	2. RFS + LDA
3. KDEF	3. LBP	3. PCA + LDA
4. JAFFE	4. HOG	4. SVM
	5. BRIEF	5. AdaBoost
		6. AdaBoost + SVM

2. Related Works

In this work, we focus on appearance approaches, in which we extract features from facial expression images by using *filters* or *transformations*, and apply classifiers to the extracted features. Our work is similar to a recent study [6] but we have more detailed analysis, including parameter sensitivity and image resolution analyses, generalization tests, and visualization of features.

On the other hand, geometric approaches use the locations of salient facial feature points such as eye corners, nose tip, and mouth corners for FER [7], [8], [9], [10]. Geometric approaches normally require a high-resolution image for accurate localization of the salient facial feature points. Shan et al. has empirically shown that geometric approaches are more suitable for FER of high-resolution facial expression images [11].

Hybrid approaches also exist and independent works show that

[1] The University of Tokyo, Bunkyo, Tokyo 113–8654, Japan
[a)] liew@ailab.t.u-tokyo.ac.jp
[b)] yairi@ailab.t.u-tokyo.ac.jp

the combined geometric and appearance features can achieve higher test accuracies [12], [13], [14]. During our literature review process, we find that FER has a trend to combine the geometric and appearance features to achieve a more robust performance. Sadeghi et al. [15] first mapped the faces to a standard shape (i.e., geometric normalization) and then extracted appearance features for FER. Happy and Routray [16] extracted appearance features from salient facial patches (i.e., geometric features) to perform FER. Similarly, Zheng [17] proposed to perform FER by using sparse SIFT feature extracted from face feature points. Eleftheriadis et al. [18] also combined geometric and appearance features to learn a Gaussian process model for FER.

Feature Descriptors: Gabor descriptor [1] is one of the most common feature descriptors that has been used for FER [12], [13], [19], [20], [21], [22], [23], [24]. Haar descriptor [2] is also popular for FER [25], [26]. Recently, LBP descriptor [3] has been adopted for FER [11], [15], [16], [18], [27] and HOG descriptor [4] has been examined for FER [28], [29], [30]. In this study, we also evaluate the performance a relatively new feature descriptor called BRIEF descriptor [5] in FER. To the best of our knowledge, we are the first to consider BRIEF descriptor in FER. Dense SIFT descriptor [31] has also been used for FER recently [32].

Classification Methods: SVM classifier [33], [34] is the most common classifier that has been applied to FER [11], [20], [21], [22], [23], [25], [27], [28], [29], [30]. Several works have reported that SVM with linear kernel produce similar test outcomes when compared to radial basis function kernel (RBF) [20], [21], [22], [23]. These consistent outcomes indicates that feature descriptors such as Gabor and Haar filters are capable of transforming the original image data into a space with a higher linear separability between image classes [*1]. On the other hand, AdaBoost [35] has also been used for FER [21], [22], [23], [26] and feature selection [11], [13], [23], [25]. Neural network approaches, including multilayer perceptron and radial basis function network, have also been explored in FER [12], [14], [24]. k-Nearest Neightbors (k-NN) classifier has also been used recently for FER [18]. Given sufficient training data, modern classifiers can normally achieve satisfactory test results. However, the best feature descriptor and classification method across different facial expression datasets remains unknown due to the lack of comparison study.

3. Facial Expression Datasets

Our experiments focus on four facial expression datasets, namely Extended Cohn-Kanade (CK+) Facial Expression Dataset [36], [37], Multimedia Understanding Group (MUG) Dataset [38], Karolinska Directed Emotional Faces (KDEF) Dataset [39], and Japanese Female Facial Expression (JAFFE) Dataset [40]. All faces in the four datasets were extracted with face detector of Computer Vision System Toolbox in MATLAB [41]. We use the detected faces directly for FER without explicitly aligning facial feature points. Littlewort et al. have reported that explicit facial feature alignment does not improve the performance significantly [23].

[*1] This process is conceptually similar to the kernel trick in SVM classifier.

Fig. 1 CK+ Facial Expression Dataset.

3.1 CK+ Dataset

CK+ Database [*2] [36], [37] is one of the most widely used facial expression databases. It has facial expression images of 210 adults aged between 18-50 years old. Participants were consisted of 69% female. All participants were requested to perform a series of facial displays with the help from an instructor. With careful selection criterion, 327 sequences were identified as one of the seven discrete facial expression, namely 'Angry', 'Contempt', 'Disgust', 'Fear', 'Happy', 'Sad', and 'Surprise'. Each sequence begins with a neutral facial expression and ends with a specific facial expression. For our comparison study, we excluded 'Contempt' class and focused on the Basic-6 facial expression [42]. We selected the first frame of all sequences as 'Neutral' images and used the last frame from all sequences as Basic-6 facial expression images. Overall, we have obtained 636 facial expression images (Angry: 45, Disgust: 59, Fear: 25, Happy: 69, Sad: 28, Surprise: 83, and Neutral: 327). **Figure 1** shows a few example images of CK+ Dataset. The square boxes were obtained from face detector in MATLAB [41]. Red, orange, yellow, green, blue, indigo, and violet colored boxes represent angry, happy, fear, neutral, sad, surprise, and disgust facial expression respectively.

It is worth noting that CK+ Database has been used for FER under different setting. Shan et al. have used the first frame of all sequence as 'Neutral' images and used the last *three* frames of all sequences as Basic-6 facial expression images for FER [11], resulting in 1,254 images (Angry: 135, Disgust: 177, Fear: 75, Happy: 207, Sad: 84, Surprise: 249, and Neutral: 327). We have also tried this setting but found that it produces unfair comparison results. The last three extracted frames are almost identical to each other and subsequently cause significant overlapping in the training and test data during cross validation. Therefore, we opted to use only the last frame from all sequences as Basic-6 facial expression images.

3.2 MUG Dataset

MUG Database [38] records facial expression images from 86 participants with Caucasian origin and aged between 20-35 years old. 59% of the participants were male. Some participants had beard or hair occlusions and 7 participants were wearing glasses. We have selected 919 facial expression images for our comparison study (Angry: 157, Disgust: 145, Fear: 118, Happy: 164, Neutral: 52, Sad: 124, and Surprise: 159). Example images are not shown here due to license agreement term.

3.3 KDEF Dataset

KDEF Database [39] records facial expression images from 140 amateur actors (70 males and 70 females) at 5 different view-

[*2] We use the term 'database' to refer to the original resources offered by third party while the term 'dataset' to refer to the images selected from the 'database' for our experiments.

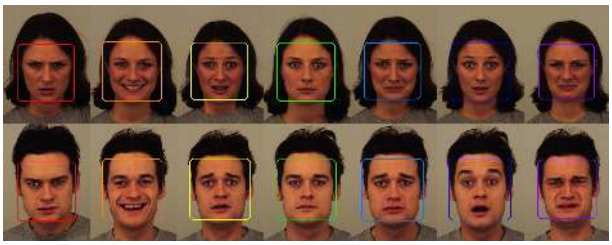

Fig. 2 KDEF Facial Expression Dataset.

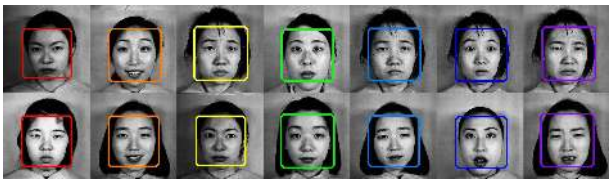

Fig. 3 JAFFE Dataset.

ing angles. All actors aged between 20-30 years old. They have no beards, no mustaches, no earrings, no eyeglasses, and mostly no visible make-up during photo sessions. For our comparison study, we only consider frontal images, resulting in 980 facial expression images (Angry: 140, Disgust: 140, Fear: 140, Happy: 140, Neutral: 140, Sad: 140, and Surprise: 140). **Figure 2** shows some example images from KDEF Dataset, where red, orange, yellow, green, blue, indigo, and violet boxes represent angry, happy, fear, neutral, sad, surprise, and disgust facial expression respectively.

3.4 JAFFE Dataset

JAFFE Database [40] contains 213 facial expression images of Basic-6 facial expressions and one neutral facial expression. All facial expression images were posed by 10 Japanese female models. We use all 213 facial expression images for our comparison study (Angry: 30, Disgust: 29, Fear: 32, Happy: 31, Neutral: 30, Sad: 31, and Surprise: 30). **Figure 3** shows some example images from JAFFE Dataset, where red, orange, yellow, green, blue, indigo, and violet boxes represent angry, happy, fear, neutral, sad, surprise, and disgust facial expression respectively.

4. Feature Descriptors

Our main goal is to identify the best feature descriptor for FER. We empirically evaluate *five* feature descriptors, namely Gabor, Haar, LBP, HOG, and BRIEF descriptors in FER. In the following, we briefly review the computational process and advantages of each feature descriptor.

4.1 Gabor Descriptor

Gabor filter theory was first formulated by Dennis Gabor in 1946's [1]. John Daugman later discovered that Gabor functions can be used to model simple cells in the visual cortex of mammalian brains [43]. This discovery reveals that Gabor filters is similar to perception in human visual system and justifies the usefulness of Gabor filters in various computer vision applications such as iris recognition [44], fingerprint matching [45], and FER [21], [22]. We illustrate a few examples of 2D Gabor filters in **Fig. 4**, where the first row shows filters with increasing frequency, second row shows filters with varying orientation, and third row shows filters with varying oscillation phase. As shown in Eq. (1) and Eq. (2), Gabor filter has compact functions that relate filter size, oscillation frequency, orientation, and oscillation phase. From Eq. (1), we can observe that the Gabor function consists of two sub-functions—Gaussian and harmonic functions, where the Gaussian sub-function (a.k.a. envelope function) is responsible in defining spatial properties (x, y, σ, γ), while the harmonic function (a.k.a. carrier function) is responsible in governing oscillation frequency (λ), orientation (θ), and oscillation phase (φ) properties of the Gabor filters.

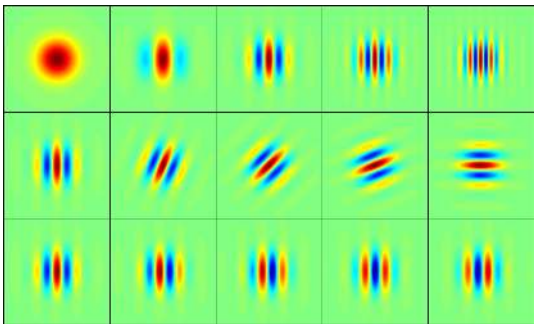

Fig. 4 Gabor filter examples. First row shows filters with increasing frequency, second row shows filters with varying orientation, and third row shows filters with varying oscillation phase.

$$g(x, y, \sigma, \gamma, \theta, \lambda, \varphi) = exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) cos\left(2\pi\frac{x'}{\lambda} + \varphi\right), \tag{1}$$

$$\begin{aligned} x' &= (x - x_o)cos\theta + (y - y_o)sin\theta, \\ y' &= -(x - x_o)sin\theta + (y - y_o)cos\theta. \end{aligned} \tag{2}$$

In our experiment, we followed Bartlett et al. [21] and set the spatial resolution (x, y) in the range of 48×48 pixels, use 8 orientations (θ = 0°, 22.5°, 45°, 67.5°, 90°, 112.5°, 135°, 157.5°), and 5 oscillation frequencies ($\lambda = 4, 4\sqrt{2}, 8, 8\sqrt{2}, 16$ pixels per cycle). Ellipticity of the Gaussian function (γ) is set to 1 and oscillation phase (φ) is set to 0. Standard deviation of the Gaussian function (σ) is set according to half-response spatial-frequency bandwidths rule in Eq. (3), where the bandwidth (b) of the Gabor function is set to 0.5.

$$\frac{\sigma}{\lambda} = \frac{1}{\pi}\sqrt{\frac{ln2}{2}} \times \frac{2^b + 1}{2^b - 1} \tag{3}$$

4.2 Haar Descriptor

Haar filter was first proposed by Papageorgiou et al. as a general framework for object recognition in 1998's [2]. Viola and Jones later popularized the Haar filter by showing its effectiveness along with integral image and cascaded classifiers in face detection problem [46]. Haar filter is a simple rectangular filter that represents the difference of sum of pixel intensities inside black and white regions. It has the key advantage of simplicity. Combined with integral image technique, Haar filter can achieve significant fast performance and make real-time face detection possible [46]. We followed Viola & Jones [46] and used the same five types of Haar filter in our experiments. **Figure 5** illustrates the five types of Haar filter, where each row shows type 1, 2, 3, 4,

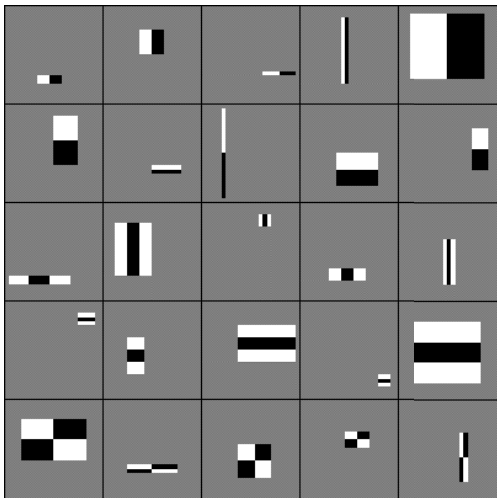

Fig. 5 Haar filter examples. The rows show five types of Haar filter with varying width and height used in our experiments.

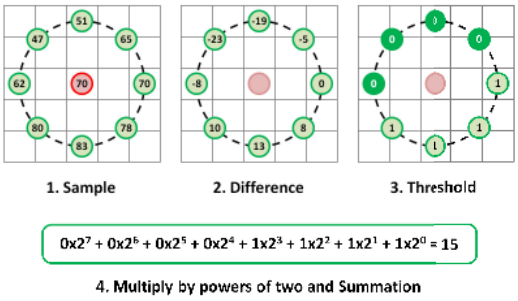

Fig. 6 Coding procedures of $LBP_{P=8,R=2}$, where P stands for number of neighbors and R stands for radial distance between the center pixel and neighboring points.

and 5 filters respectively with different sizes at different locations. We set the black and white regions to have the same size. We also increased the size of black and white regions with a step size of 2 pixels. As a result, we extracted 162,336 Haar values from a 48×48 image.

4.3 LBP Descriptor

Local Binary Pattern (LBP) descriptor proposed by Ojala et al. [3] is a powerful feature descriptor for image classification. Compared to Gabor or other image filters, LBP has the advantages of computational simplicity and robustness against illumination variations. LBP encodes information of local patterns such as edges, lines, and spots in each pixel. Equations (4) and (5) summarize the encoding processes in a compact form while **Fig. 6** illustrates the encoding procedures for a $LBP_{P=8,R=2}$ operator in details. More precisely, LBP coding for every pixel is computed as follows:

(1) Sample neighbor points, where P defines the number of neighboring points and R defines the radial distance between the center pixel to neighboring points.
(2) Compute the difference of pixel values between center pixel and neighbor points.
(3) Threshold the computed differences at zero.
(4) Multiply the thresholded values with power of two consequently and sum all the values.

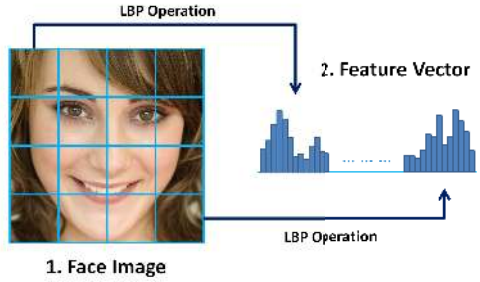

Fig. 7 Face description with LBP descriptor. Face image [48] is first divided into cells and followed by LBP operation on each cell. Histogram of each cell is concatenated into a feature vector.

It is worth noting that the $LBP_{8,2}$ operator produces 2^P possible binary patterns and it has been shown that some binary patterns contain more information than others [3]. Ojala et al. named these binary patterns as uniform patterns, where they contain at most two bitwise transitions from 0 to 1 or vice versa (considering binary pattern in circular). For instance, 00000001, 11001111, and 11110011 are uniform patterns while 10100000, 00011101, and 11001100 are non-uniform patterns. Ojala et al. found that about 90% of all binary patterns are uniform and they proposed to accumulate all non-uniform patterns into single bin. Therefore, the original $2^8 = 256$ bins is reduced to 59 bins. We followed this practice in our experiments, similar to previous facial expression study [11] and face recognition study [47].

After LBP coding of every pixels, LBP values of pre-defined cells (**Fig. 7**) are stored in histogram form and eventually concatenated into a 1D feature vector. More precisely, we start with a window size of 12×12 pixels, compute the histogram of LBP values inside the window, and continue the process by shifting the current window to the right hand side by 3 pixels. If the right corner is reached, window will be shifted to the bottom side by 3 pixels and be restarted from the left hand side. Normalization of histograms is optional. In our case, we did not carry out normalization since all the considered windows are in the same size.

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c) \cdot 2^p \tag{4}$$

$$s(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

4.4 HOG Descriptor

Histogram of Oriented Gradients (HOG) feature was first described by Dalal et al. for human detection [4]. While being similar to Scale-invariant Feature Transform (SIFT) descriptor [49], HOG represents dense coding of image and have some unique details such as using different number of histogram bins and different image block size [4]. HOG descriptor can be computed in the following five basic steps:

(1) Gradient computation—image is convoluted with two Sobel filters, i.e., $[-1,0,1]$ and $[-1,0,1]^T$, to form horizontal and vertical gradient maps. Following the common practice, smoothing and gamma normalization are omitted [4].
(2) Magnitude and orientation computation—magnitude and

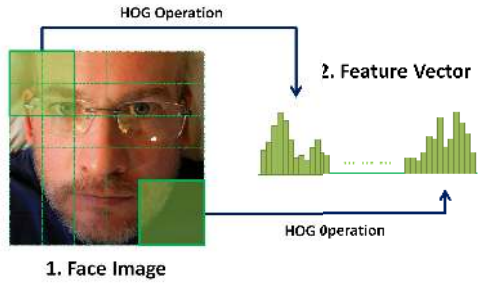

Fig. 8 Face description with HOG descriptor. Face image [50] is first divided into cells and followed by quantization of gradient orientation on each cell. Histogram of each block of four adjacent cells is then locally normalized and concatenated into a full feature vector.

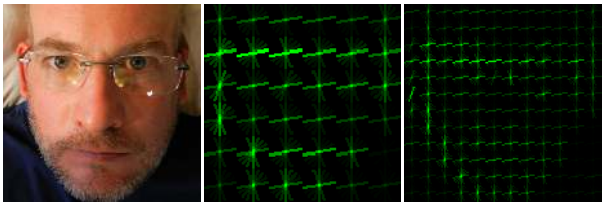

Fig. 9 Visualization of HOG descriptors with divided cells. (a) Face image [50]; (b) HOG visualization of 6×6 cells; (c) HOG visualization of 12×12 cells.

orientation maps are computed based on the obtained horizontal and vertical gradient maps. Taking dx and dy as gradient value of pixel in the horizontal map and vertical maps, magnitude and orientation are calculated as: $Magnitude = \sqrt{(dx)^2 + (dy)^2}$ and $Orientation = tan^{-1}(\frac{dy}{dx})$.

(3) Cell division—image is then divided into smaller cells. For example, we are using face images with size of 48×48 pixels in our experiments. These images are divided into 6×6 cells where each cell has size of 8×8 pixels (**Fig. 8**).

(4) Cell quantization—the orientation values of each cell is quantized in histogram form with 9 orientation bins, where the magnitude represents voted weights, and we interpolate votes bi-linearly between neighboring bin center.

(5) Block normalization—four adjacent cells form a block with 16×16 pixels (every block has 50% overlapping with the adjacent block). Orientation histograms of each block are locally normalized and concatenated into a feature vector.

Figure 9 (a) shows an example face image in size of 48×48 pixels; Fig. 9 (b) shows the corresponding HOG descriptors under 6×6 cells configuration (with cell size of 8×8 pixels); and Fig. 9 (c) shows the corresponding HOG descriptors under 12×12 cells configuration (with cell size of 4×4 pixels). Overall, the standard HOG descriptor used in our experiments has a dimension of (5×5×4×9 + 11×11×4×9) = 5,256.

4.5 BRIEF Descriptor

Binary Robust Independent Elementary Features (BRIEF) descriptor [22], [51] was first proposed for image matching with random forest [22] and random ferns classifiers [51]. BRIEF descriptor has the lowest computational cost among Gabor, Haar, LBP, and HOG descriptors because it only performs simple binary comparison test and uses Hamming distance (instead of Euclidean or Mahalanobis distance) for image classification [52].

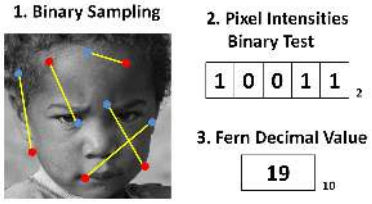

Fig. 10 Face image [53] and its BRIEF descriptor. Five random pixel pairs are selected, followed by binary tests on pixel intensities, and conversion from binary code into decimal value.

BRIEF descriptor has two setting parameters – the number of binary pixel pairs and binary threshold. We used five binary pixel pairs and zero binary threshold in all our experiments. **Figure 10** illustrates computation process of a BRIEF descriptor. Firstly, five binary pixel pairs are randomly chosen from a given image. Secondly, binary pixel pairs are subjected to binary test,

$$f_n = \begin{cases} 1, & \text{if}(I_a - I_b) \geq 0 \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

where I_a and I_b are the intensity values of red circle pixel and blue circle pixel in Fig. 10, respectively. Thirdly, the binary values of five binary pixel pairs are summarized in binary code and subsequently converted into decimal value, which can be summarized in the following form,

$$F = \sum_{n=1}^{5} f_n \cdot 2^{n-1} \,. \quad (7)$$

In all our experiments, we use 10,000 BRIEF values for FER.

5. Classification Methods

After feature extraction, we applied six different classifiers with the feature descriptors. We first considered k-NN classifier, followed by LDA classifier. Thirdly, we used Principal Component Analysis (PCA) to reduce the size of feature descriptors and used LDA for FER. Fourthly, we used SVM and AdaBoost classifier for FER. Last but not least, we used AdaBoost for feature selection and SVM for FER.

We used two-class classifiers in all experiments. Specifically, we built seven one-vs.-all classifiers that are responsible in recognizing seven facial expressions, namely 'Angry', 'Disgust', 'Happiness', 'Fear', 'Neutral', 'Sad', and 'Surprise'. We used common voting practice and combined seven test results in the end [11], [23]. For example, an image identified as positive by 'Angry' classifier will get +1 point for 'Angry' label and −1 point for other labels. Similarly, the same image identified as negative by 'Neutral' classifier will get −1 point for 'Neutral' label and +1 point for other labels. In order to avoid possible classification ties, each label is initiated with a random positive number that is smaller than 1. Label with the highest points after combining seven voting results will be elected as the final test label.

5.1 k-Nearest Neighbors (k-NN)

Given a test data in a feature descriptor space, 1-NN classifier tries to find a training data that is the closest to the test data, and consider the label of that training data as the test label. Similarly,

given a test data in a feature descriptor space, k-NN classifier tries to find k training data that are the closest to the test data, and consider the label with the largest occurrence as the test label. In our experiments, we considered Euclidean distance and used exhaustive search. Denoting a particular training data as x_k and test data as x, the squared Euclidean distance between the training and test data can be computed as $d_k = (x_k - x)^T(x_k - x)$.

5.2 Linear Discriminant Analysis (LDA)

LDA assumes facial expression images to follow multivariate normal distribution model and all classes C have the same covariance matrix. During training process, LDA estimates the means μ_c and covariance matrix Σ_c of every class c and tries to look for an optimal linear boundary between classes [54]. Given a test data $x \in \mathbb{R}^{D\times 1}$, the test label can be predicted by maximizing

$$\hat{c} = \arg\max_{c=1,\ldots,C} \hat{P}(c|x) \,. \tag{8}$$

The posterior probability in Eq. (8) can be computed by using Bayesian rule

$$\hat{P}(c|x) = \frac{P(x|c)P(c)}{P(x)} \,, \tag{9}$$

where the $P(c)$ is class prior, the $P(x)$ is a normalization constant, and the likelihood can be computed by

$$P(x|c) = \frac{1}{\sqrt{(2\pi)^D|\Sigma_c|}} \exp\Big(-\frac{1}{2}(x-\mu_c)^T\Sigma_c^{-1}(x-\mu_c)\Big) \,. \tag{10}$$

In practice, the number of our training data is smaller than the dimensions of feature descriptors. This condition would trigger singularity issue when we invert the covariance matrix during the training process. Moreover, the inversion of covariance matrix has large computational load due to the large feature descriptors (about 90,000 for Gabor descriptor and more than 160,000 for Haar descriptor). To maintain simplicity of LDA as well as comparison with PCA at later stage, we randomly selected 200 features from each descriptors for FER.

While LDA is always used interchangeably with Fisher LDA (FLDA), we follow the naming convention in Refs. [54], [55], [56] to avoid confusion. We consider LDA as a *classification* method while FLDA as a *dimensionality reduction* method. Formally, LDA fits data with multivariate normal function with the assumption that each class shares a same covariance matrix. On the other hand, FLDA tries to maximize classes' separability by finding an optimal linear projection. Under this point of view, 'LDA' mentioned in the previous facial expression studies [11], [23] are in fact FLDA method. For more details, please refer to Chapter 4.2.2 (LDA) and Chapter 8.6.3 (FLDA) in Ref. [54] or online resources [55], [56].

5.3 Principal Component Analysis (PCA)

In the previous section, we used random feature selection (RFS) to reduce the number of features in order to avoid singularity issue in LDA classifier and save computational cost. In this section, we performed dimensionality reduction more systematically by using Principal Component Analysis (PCA) [57].

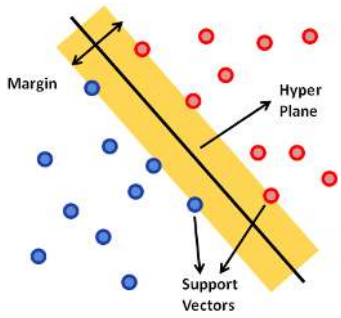

Fig. 11 SVM classifier illustration. SVM find an optimal hyper-plane that maximizes the margin area in between two classes of data.

In theory, PCA finds an orthogonal projection plane in a lower dimensional space that minimize the data's projection error. From another point of view, PCA also maximizes the variances between data while projecting data to a lower dimensional space. Principal components can be computed by using eigen decomposition or singular value decomposition methods. In practice, the first few principal components normally capture most of the information in the original data. We used the first 200 principal components (retaining more than 98% of information) for LDA classification. In our experiments, we used power method of PCA [58] to reduce the computational load and speedup performance.

5.4 Support Vector Machine (SVM)

SVM was originally proposed by Vapnik and Lerner in 1963's [33] and the concept of soft margin in SVM was formulated by Cortes and Vapnik in 1995's [34]. Unlike LDA, SVM assume no prior knowledge about the distribution of the samples. As shown in **Fig. 11**, SVM looks for an optimal hyper-plane that maximizes the margin area between the two classes' closest training sample points (support vectors).

In practice, we have to decide a kernel function, such as linear, polynomial, or RBF for SVM [59], [60] and there is no analytical way to decide which kernel function works the best for any particular datasets. In this study, we focus on SVM with linear kernel (linear SVM) for two main reasons. Firstly, Littlewort et al. and Shan et al. have reported that SVM with radial basis function (RBF SVM) does not perform significantly better than linear SVM [11], [23]. These outcomes indicate that facial expression datasets are highly linear-separable upon feature transformation. Secondly, linear SVM has only one tuning parameters (soft margin) and has a lower computational cost than RBF SVM.

With linear SVM, we tuned the soft margin parameter by performing a grid search over range of $10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}$ and identified the parameter that leads to the best performance. Datasets were normalized prior to the classification. We used `svmtrain` and `svmclassify` functions in MATLAB Statistics Toolbox [61]. We also used sequential minimal optimization (SMO) method [62] since it performs significantly faster and produces similar results with the quadratic programming method.

5.5 Adaptive Boosting (AdaBoost)

AdaBoost was formulated by Freund and Schapire in 1995's [35], [63]. While SVM tries to find the *training samples* that are the most difficult to classify (support vector) and form

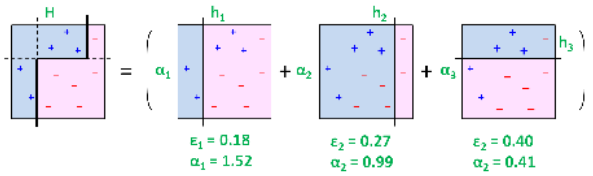

Fig. 12 AdaBoost classifier illustration. AdaBoost classifier selects the best feature for each weak classifier h at one time, and then update the sample weights α based on the local classification error ϵ.

optimal hyper plane based on these training samples, AdaBoost looks for the best *training features* that are useful for its weak classifiers. AdaBoost iteratively select the best training feature on each training cycle. After each feature selection, sample weights are re-adjusted according to the local classification error. The same process repeats until a certain number of features are selected. Weak classifiers, normally linear decision stumps with only threshold and polarity parameters, are then combined together to form a strong classifier. Mathematically, the linear decision stump classifier can be written as

$$h_t(x, p, \theta) = \begin{cases} +1, & \text{if } p \cdot f(x) < p \cdot \theta\,, \\ -1, & \text{otherwise}\,. \end{cases} \tag{11}$$

where $f(x)$ is a feature descriptor, θ represents a threshold value, and p represents the direction of the inequality. We illustrate AdaBoost algorithm graphically in **Fig. 12** and summarize the essential steps in Algorithm 1.

Algorithm 1 AdaBoost Algorithm

Step 1: Denoted as (x_i, y_i), training samples consist of a vectorized facial expression image x_i and training label $y_i = \{+1, -1\}$.

Step 2: Given P positive and N negative training data, set the weights of positive and negative samples to $w_i = \frac{1}{P}$ and $w_i = \frac{1}{N}$ respectively.

For $t = 1, \ldots, T = 80$:

Step 3: Normalize the weights of all training samples, $w_i \leftarrow \frac{w_i}{\sum_{i=1}^{k} w_i}$.

Step 4: By using Eq. (11) with $p = 1$ or $p = -1$ and different θ, compute the error rate ϵ_t of each decision stump $h_t(x)$.

Step 5: Select the decision stump $h_t(x)$ that has lowest error rate.

Step 6: Compute weight of selected decision stump, $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$.

Step 7: Update training sample weights, $w_i \leftarrow w_i \cdot \exp(-\alpha_t y_i h_t(x_i))$.

Step 8: The final strong classifier is $H(x) = \text{sign}[\sum_{t=1}^{T=80} \alpha_t h_t(x)]$.

5.6 AdaBoost with SVM (AdaBoostSVM)

AdaBoost is a special classifier as it performs feature selection and classification simultaneously by combining numerous weak classifiers in an additive manner. Useful features are identified by weak classifiers during the training process. A number of weak classifiers and features are then combined to form a strong classifier to carry out the image classification. Since SVM and AdaBoost design a robust classifier from different perspectives, i.e., SVM selects *training samples* to be support vectors while AdaBoost identifies the best *training features* for weak classifiers, we expect that the integration of SVM and AdaBoost will lead to an even stronger classifier. One possible way is to use SVM as the weak classifier in AdaBoost. However, this would possibly increase both the training and test times in practice. Moreover, there is no guarantee that SVM will outperform decision stump since each weak learner only considers one feature. In contrast, we use AdaBoost to select the best training features and use SVM for classification. Specifically, all the features selected by all weak classifiers, i.e., decision stump, during the AdaBoost training process are adopted as the features for training the SVM classifier. Our concept is similar to Refs. [11], [13], [23], [25] but we fixed the number of selected features at 80 in our studies, which is close to the number of support vectors in our SVM classifiers. Note that we could only determine a very rough estimate of the number of support vectors since we are performing multiple one-vs.-all experiments in 20 validation cycles.

Table 2 Test accuracies of k-Nearest Neighbors (CK+ Dataset).

Descriptors	1-NN	5-NN	10-NN	20-NN
Gabor	57.7 ± 5.4	50.5 ± 7.3	59.5 ± 5.7	60.9 ± 5.6
Haar	52.7 ± 4.9	41.1 ± 6.5	47.3 ± 5.1	47.3 ± 5.1
LBP	**60.2 ± 5.1**	**58.1 ± 7.7**	60.5 ± 6.8	59.3 ± 6.3
HOG	56.5 ± 4.4	52.3 ± 5.9	**66.4 ± 4.3**	**68.4 ± 3.6**
BRIEF	55.6 ± 5.4	51.1 ± 6.6	57.7 ±5.6	57.6 ± 6.0

6. Classification Results

We report experiment results of each feature descriptors with six classifiers: k-NN, RFS+LDA, PCA+LDA, SVM, AdaBoost, and AdaBoostSVM classifiers in this section. We carried out repeated random sub-sampling validation in all our experiments. Specifically, we randomly selected 90% of all images for training and used the remaining 10% images for testing. We repeated this process for 20 times and report the mean and standard deviation of test accuracies in the followings. In the meantime, we focus our discussion on experiment results of CK+ Dataset. In Section 11, we will generalize our test results to the remaining datasets (MUG, KDEF, JAFFE Datasets) and a combined dataset (CK+ & MUG & KDEF & JAFFE).

6.1 k-Nearest Neighbors (k-NN)

Table 2 summarizes the test accuracies of five feature descriptors with k-NN classifiers applied to the CK+ Dataset. We tested performance of 1-NN, 5-NN, 10-NN, and 20-NN classifiers with an exhaustive search. While the overall performance is the worst among all classifiers, we found an interesting pattern in k-NN classification results. We find that when the number of nearest neighbors is small, LBP descriptor tends to produce the best results. On the other hand, when the number of nearest neighbors is large, we find that HOG descriptor tends to produce the best results. Moreover, the larger the number of nearest neighbors, the higher the test accuracies. However, a larger number of nearest neighbors would lead to a higher computational cost. In short summary, we considered 20 as the largest number of nearest neighbors in this experiment and identified HOG descriptor as the best feature descriptor for FER in k-NN classifier.

Table 3 Test accuracies of Random Feature Selection+LDA and PCA+LDA (CK+ Dataset).

Descriptors	RFS+LDA	PCA+LDA
Gabor	79.4 ± 5.2	82.7 ± 3.4
Haar	80.8 ± 4.9	83.4 ± 4.2
LBP	66.4 ± 6.0	86.4 ± 2.6
HOG	**82.5 ± 4.1**	**90.9 ± 3.2**
BRIEF	67.2 ± 5.9	83.7 ± 3.4

Table 4 Test accuracies of Linear SVM and AdaBoost (CK+ Dataset).

Descriptors	Linear SVM	AdaBoost	AdaBoostSVM
Gabor	83.6 ± 3.4	81.1 ± 5.8	79.5 ± 4.0
Haar	80.2 ± 3.5	78.0 ± 4.1	74.7 ± 4.6
LBP	86.0 ± 4.0	81.2 ± 3.9	82.9 ± 4.6
HOG	**91.2 ± 3.2**	**85.7 ± 3.0**	**85.9 ± 4.0**
BRIEF	83.2 ± 3.4	79.7 ± 4.4	78.9 ± 4.9

6.2 Linear Discriminant Analysis (LDA)

Table 3 summarizes the test accuracies of five feature descriptors with RFS+LDA and PCA+LDA classifiers applied to the CK+ Dataset. First of all, we observe that the performances of both RFS+LDA and PCA+LDA classifiers are better than all k-NN classifiers. Secondly, as expected, PCA performs better than RFS as PCA reduces the size of feature descriptors while capturing the most important information systematically. Last but not least, we find that HOG descriptor produces the best test accuracies in both RFS+LDA and PCA+LDA classifiers.

6.3 Support Vector Machine (SVM) and AdaBoost

Table 4 summarizes the test accuracies of five feature descriptors with SVM, AdaBoost, and AdaBoostSVM classifiers applied to the CK+ Dataset. We observe that once again, HOG descriptor produces the best test accuracies in all the three classifiers. Here, we examine the classifiers more carefully. First, we find that linear SVM classifier produces the best test accuracy among all the six classifiers in our experiments. Second, out of our expectation, we find that AdaBoost classifier performs slightly worse than linear SVM classifier. Third, contrary to previous studies [13], [23], we find that in most cases, AdaBoostSVM classifier produces worse result than both linear SVM and AdaBoost classifiers. We believe that this discrepancy is caused by the uses of different experiment datasets, different tuning practices in SVM classifier, and different weak learners in AdaBoost classifier.

In addition to the test accuracies, we also analyzed the influence of the soft margin (hence the number of support vectors) on the SVM performance, as well as the influence of the number of weak learners on the AdaBoost performance. We focused on the HOG and BRIEF descriptors in order to save computational time. In the experiments, we varied the soft margin parameter over the range of $10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}, 10^{4}, \infty$ and found that all the settings produce similar test accuracy results (within ±0.5%). This indicates that the facial expression images are highly linearly separable upon the feature extraction or transformation.

In the AdaBoost experiments, we varied the number of weak learners over the range of 40, 80, 120, 160, 200. **Figure 13** shows the training and test accuracies of AdaBoost classifier at different number of weak learners (decision stump) with HOG (left) and BRIEF descriptors (right). We can observe that both descriptors start to achieve 100% training accuracies when the number of weak learners equal to 80. We also find that the test performances of HOG and BRIEF descriptors start to saturate when the number of weak learners are equal to 120 and 160 respectively. While we fixed the number of weak learners to 80 in the (default) comparison experiments, the differences between default and the best test accuracies are not large (less than 1% in HOG case and less than 2% in BRIEF case). Nevertheless, these results suggest that we could increase the number of weak learners from 80 to 120 (in the HOG case) and to 160 (in the BRIEF case) in practice in order to achieve more accurate performance.

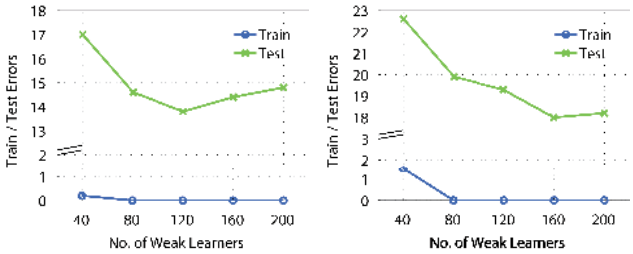

Fig. 13 Training and test errors of AdaBoost classifier at different number of weak learners with HOG (left) and BRIEF descriptors (right).

6.4 Overfitting Consideration

Since the CK+ Dataset that we are using is relatively small when compared to the dimensions of feature vectors, overfitting issue should be analyzed in order to consolidate the classification results. In addition to the previously mentioned repeated subsampling validation procedures, we further performed an overfitting test by plotting the training error and validation error as a function of the size of the training dataset. Since performing the experiments with all feature descriptors is time-consuming, we focused the overfitting test on HOG and BRIEF descriptors with SVM classifier. We chose HOG descriptor because it has the best test performance among other feature descriptors. We chose BRIEF descriptor because it has an out-of-expectation result in Section 10, where its classification performance outperform LBP and HOG descriptors when the image resolution is lower than 24×24 pixels. Note that the dimensions of both HOG and BRIEF descriptors are larger than the size of the training dataset. On the other hand, we chose SVM classifier because it does not perform feature selection and considers the full feature vector during the image classification.

Figure 14 shows the training and test accuracies of SVM classifier at different percentage of training data with HOG (left) and BRIEF descriptors (right). Note that all the experiment settings remain the same. We randomly sub-sample 10% of the full data for test, and vary the percentage of the remaining data to train the SVM. We repeated all experiments for 20 times and report the mean error results. From the plots, we can observe that both descriptors always have 0% training errors and have *decreasing* test errors, indicating that *no* overfitting occurs in our experiments. Moreover, the results suggest a lack of training data. With more training data, we expect an increase of training errors and further decrease of test errors.

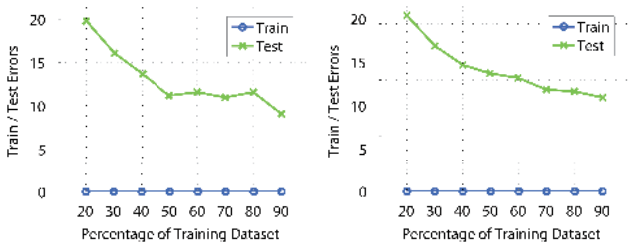

Fig. 14 Training and test errors of SVM classifier at different percentage of training data with HOG (left) and BRIEF descriptors (right).

6.5 Discussion

In this section, we focus on the basic version of all feature descriptors. From this point of view, we observe that most of the time HOG descriptor produces the best test accuracies (except 1-NN and 5-NN classifiers). Compared to other descriptors, HOG descriptor is more powerful as it considers a few important feature extraction processes, such as the uses of image gradient, overlapping blocks, and histogram normalization.

The performance of Gabor descriptor in our experiments is also slightly different with works of Bartlett et al., possibly due to the use of different dataset [*3] and different cross-validation approach [21]. Here, we use 20-round repeated random sub-sampling validation while Bartlett et al. use leave-one-subject-out cross validation.

7. Parameters Sensitivity Analysis

As described in Section 4, all Gabor, Haar, LBP, HOG, and BRIEF descriptors have a number of parameter choices that may significantly affect our experiment performances. We varied a few important parameters during the feature extraction processes in our experiments and report their sensitivities in this section. For all feature descriptors, we focus on PCA+LDA, SVM, and AdaBoost classifiers. All other experiment settings remain unchanged unless mentioned otherwise. Note that some results in this section are not exactly the same with the results in the previous section because we are using repeated random sub-sampling validation in our experiments.

7.1 Gabor Parameters

As shown in Eq. (1), Gabor filter has five parameter choices, where the differences in oscillation frequency (λ) and orientation (θ) can result in very different filters. In Section 6, we extracted Gabor descriptors by using 5 oscillation frequencies ($\lambda = 4, 4\sqrt{2}, 8, 8\sqrt{2}, 16$ pixels per cycle) and 8 orientations ($\theta = 0°, 22.5°, 45°, 67.5°, 90°, 112.5°, 135°, 157.5°$). Here, we denote this setting as $f = 5$ and $o = 8$. We varied these two parameters individually by first using 3 oscillation frequencies ($\lambda = 4, 8, 16$ pixels per cycle) or 1 oscillation frequency ($\lambda = 8$ pixels per cycle). Then, we used 4 orientations ($\theta = 0°, 45°, 90°, 135°$) or 2 orientations ($\theta = 0°, 90°$) to extract Gabor descriptors.

We summarized the classification results of the three classifiers in **Table 5**, where f and o represent the number of oscillation frequencies and orientations mentioned in the last paragraph. From the first three rows, we can observe that all three classification results improve steadily, indicating that increasing the number of oscillation frequencies has a positive impact towards the performance. From the last three rows, we can also observe that all three classification results improve steadily (from bottom to the third row), indicating that increasing the number of orientations has a positive impact towards the performance. In a short summary, these two parameters have high sensitivity towards the classification performance. While further increase the numbers of both parameters might continue to improve the classification performance, we limit $f = 5$ and $o = 8$ in our experiments in order to save computational cost.

[*3] We use the latest version of CK+ Database.

Table 5 Test accuracies of PCA+LDA, SVM, and AdaBoost (CK+ Dataset) with varied oscillation frequencies and orientations when extracting Gabor descriptor. Please refer to the text for more details.

Gabor	PCA+LDA	SVM	AdaBoost
f = 1, o = 8	77.9 ± 4.4	73.3 ± 6.7	75.3 ± 6.2
f = 3, o = 8	83.2 ± 5.9	80.1 ± 6.1	78.8 ± 4.7
f = 5, o = 8	**83.3 ± 4.7**	**83.4 ± 5.5**	**79.8 ± 4.4**
f = 5, o = 4	81.3 ± 4.1	78.8 ± 6.5	79.0 ± 4.6
f = 5, o = 2	78.4 ± 4.3	73.8 ± 5.7	77.0 ± 5.3

Table 6 Test accuracies of PCA+LDA, SVM, and AdaBoost (CK+ Dataset) with varied step sizes and filter types when extracting Haar descriptor. Please refer to the text for more details.

Haar	PCA+LDA	SVM	AdaBoost
s = 4, t = 5	82.9 ± 5.4	75.9 ± 6.3	64.9 ± 5.9
s = 3, t = 5	**84.0 ± 5.3**	78.1 ± 6.0	70.6 ± 6.5
s = 2, t = 5	83.7 ± 4.2	78.8 ± 5.7	76.7 ± 5.3
s = 2, t = 4	83.1 ± 4.6	79.4 ± 5.8	**77.6 ± 6.3**
s = 2, t = 2	83.1 ± 5.0	**79.7 ± 5.0**	77.5 ± 5.4

7.2 Haar Parameters

Compared to Gabor filter, Haar filter has less parameter choices. In Section 6, we extracted Haar descriptors with 5 types of Haar filter and with a step size of 2 pixels. Here, we denote this setting as $s = 2$ and $t = 5$. Similar to the Gabor filter case, we varied these two parameters individually by first using step sizes of 4 or 3. After that, we used the first 4 types or the first 2 types of Haar filter (please refer to Fig. 5) to extract Haar descriptors.

We summarized the classification results of the three classifiers in **Table 6**, where s and t represent the number of step size and number of types of filter mentioned in the last paragraph. From the first three rows, we can observe that all three classification results improve steadily, indicating that increasing the number of step size has a negative impact towards the performance. Out of our expectation, from the last three rows, we observe that using more types of Haar filter indeed does not improve the results of all three classifiers. While more types of Haar filter could be helpful in differentiating face and non-face images [46], we find that using the first two types of Haar filter (the simplest Haar filter) is the best for FER. In a short summary, one should use a step size of 2 pixels of and use the first two types of Haar filter for FER [*4].

[*4] Since we perform this analysis at the later stage of our comparison study, we keep the results of the original setting, i.e., $s = 2$ and $t = 5$ in all other sections. Note that this does not affect our conclusion that HOG is the best descriptor for FER since different t produce similar test results.

Table 7 Test accuracies of PCA+LDA, SVM, and AdaBoost (CK+ Dataset) with varied neighboring points and radial distances when extracting LBP descriptor. Please refer to the text for more details.

LBP	PCA+LDA	SVM	AdaBoost
p = 4, r = 2	86.8 ± 4.7	86.5 ± 5.2	77.8 ± 4.8
p = 6, r = 2	**89.2 ± 3.7**	**88.3 ± 3.9**	79.3 ± 4.2
p = 8, r = 2	85.8 ± 4.5	85.4 ± 5.4	79.4 ± 5.0
p = 8, r = 3	88.9 ± 4.0	85.9 ± 5.0	78.9 ± 5.5
p = 8, r = 4	87.4 ± 4.2	83.9 ± 4.7	**79.5 ± 5.1**

Table 8 Test accuracies of PCA+LDA, SVM, and AdaBoost (CK+ Dataset) with varied cell sizes and combination when extracting HOG descriptor. Please refer to the text for more details.

HOG	PCA+LDA	SVM	AdaBoost
c = 8	84.7 ± 6.2	84.1 ± 4.4	80.8 ± 6.4
c = 6	90.6 ± 3.5	89.1 ± 3.9	84.1 ± 5.0
c = 4	**91.6 ± 3.6**	90.7 ± 3.9	**86.0 ± 3.9**
c = 8, c = 6	89.8 ± 3.6	89.8 ± 3.5	83.8 ± 5.4
c = 8, c = 4	91.2 ± 3.2	91.3 ± 4.0	85.2 ± 5.0
c = 6, c = 4	90.6 ± 3.7	90.6 ± 4.5	85.3 ± 5.8
c = 8, c = 6, c = 4	91.2 ± 4.1	**91.6 ± 3.7**	84.8 ± 4.3

7.3 LBP Parameters

LBP descriptor has two parameter choices. In Section 6, we extracted LBP descriptors with 8 neighboring pixels at radial distance of 2 pixels. Here, we denote this setting as $p = 8$ and $r = 2$. Similar to the previous cases, we varied these two parameters individually by first using 6 or 4 neighboring pixels. After that, we used the radial distance of 3 or 4 pixels to extract LBP descriptors.

We summarized the classification results of the three classifiers in **Table 7**, where p and r represent the number neighboring pixels and radial distance. From the first three rows, we can observe that all three settings produce the best results when $p = 6$. From the last three rows, we find that PCA+LDA and SVM classifiers perform the best at $r = 2$ and $r = 3$ while AdaBoost classifier perform similarly and appears to be independent of radial distance. In contrast to a previous study [11] and our default parameter settings, one should use 6 neighboring pixels and different range of radial distance depending on the classifiers [*5].

7.4 HOG Parameters

As described in Section 4, HOG descriptor is compact and has only one parameter choice—its cell size. In Section 6, we extracted HOG descriptors with cell sizes of 8 and 4, and stacked the two feature vectors together eventually. Here, we denote this setting as $c = 8$ and $c = 4$. Different with the previous cases, we varied the cell size and analyze the test accuracies by using individual feature vector or stacked feature vectors.

We summarized the classification results of the three classifiers in **Table 8**, where c represents the cell size in pixel values. The rows with multiple c represent stacked feature vectors with different cell sizes. From the first three rows, we can observe that all three classification produce the best classification results when $c = 4$. When $c = 4$, the resulting number of blocks and dimension of HOG descriptor is larger and can potentially capture more detailed edge information from the facial expression images. From the last four rows, we find that all classifiers produce similar (PCA+LDA and AdaBoost) or slightly better (SVM) test accuracies than HOG descriptor with one cell size. Note that we use $c = 8$ and $c = 4$ as our default experiment settings, which has similar outcomes to the best test accuracies in the Table 8.

[*5] Similarly, we keep the results of the original setting, i.e., $p = 8, r = 2$ in all other sections. This does not affect our conclusion since the LBP descriptor with the latest settings still does not outperform HOG descriptor.

Table 9 Test accuracies of PCA+LDA, SVM, and AdaBoost (CK+ Dataset) with varied number of binary pixel pairs and descriptors when extracting BRIEF descriptor. Please refer to the text for more details.

BRIEF	PCA+LDA	SVM	AdaBoost
b = 2, f = 10,000	83.1 ± 5.8	82.6 ± 5.5	80.6 ± 4.2
b = 5, f = 10,000	**83.6 ± 4.8**	**83.0 ± 4.4**	79.5 ± 5.0
b = 8, f = 10,000	82.9 ± 5.3	83.0 ± 4.9	**80.9 ± 4.0**
b = 5, f = 5,000	82.7 ± 5.1	82.2 ± 5.4	77.5 ± 5.6
b = 5, f = 2,000	80.1 ± 5.5	80.9 ± 5.2	76.4 ± 5.2

Table 10 Test accuracies of PCA+LDA, SVM, and AdaBoost (CK+ Dataset) with LBP and LBP Histogram Fourier descriptors.

	PCA+LDA	SVM	AdaBoost
LBP	**86.4 ± 2.6**	**86.0 ± 4.0**	**81.2 ± 3.9**
LBP-HF	82.1 ± 6.2	80.6 ± 4.4	73.6 ± 4.9

7.5 BRIEF Parameters

BRIEF descriptor has two parameter choices—the number binary pixel pairs and number of BRIEF values. In Section 6, we extracted 10,000 BRIEF values with 5 binary pixel pairs. Here, we denote this setting as $b = 5$ and $f = 10{,}000$. Similar to the Gabor, Haar, and LBP cases, we varied these two parameters individually by first using 2 or 8 binary pixel pairs while extracting 10,000 BRIEF values. After that, we fixed binary pixel pairs to 5 and extracted 5,000 or 2,000 BRIEF values.

We summarized the classification results of the three classifiers in **Table 9**, where b and f represent the number of binary pixel pairs and number of BRIEF values. From the first three rows, we can observe that PCA+LDA and SVM perform the best when $b = 5$ while AdaBoost performs the best when $b = 8$. From the last three rows, we can observe that the test accuracies deteriorate with decreasing number of BBIEF values. In general, one should use a larger f if the computational cost is not an issue. On the other hand, we find that using $b = 5$ has a good balance in between computational cost and accuracy performance (except for the AdaBoost case, but nevertheless the difference is small).

7.6 Advanced Variants

In addition to the analysis of feature parameters, it is also worth testing some advanced variants of the classic feature descriptors. We further compared the FER performance by extracting LBP Histogram Fourier (LBP-HF) descriptor [64] and Gaussian BRIEF descriptor (G-BRIEF) [65]. The LBP-HF descriptor is a rotation invariant descriptor computed from Fourier transforms of LBP histogram. It has been reported that LBP-HF descriptor outperforms classic LBP in texture classification and face recognition tests [64]. **Table 10** summarized the test results of the classic LBP and LBP-HF descriptors. Based on our experiments, we find that the LBP-HF descriptor produce worse test results when compared to the classic LBP descriptor. We speculate that the LBP-HF descriptor will perform better on datasets with in-plane rotations. Since our current datasets always have the same head

pose, the LBP-HF descriptor does not prove to be very useful.

Table 11 summarizes the test results of the classic BRIEF and G-BRIEF descriptors. Based on our experiments, we find that the G-BRIEF descriptor perform slightly better than the classic BRIEF descriptor only in the case of SVM classifier. It has been reported that G-BRIEF is good at dealing with image Gaussian noise and partial occlusions. We speculate that G-BRIEF will performs better if there is Gaussian noises or partial occlusions in our FER datasets. To this end, we find that HOG descriptor is still the best descriptor for FER in consideration of the test accuracy.

8. Confusion Matrices

We present confusion matrices of HOG descriptor applied with SVM classifier in order to identify the weak points of feature descriptors. **Table 12** presents the confusion matrix of HOG descriptor with linear SVM classifier applied to the CK+ Dataset, with 'An', 'Di', 'Fe', 'Ha', 'Sa', 'Su', and 'Ne' represent 'Angry', 'Disgust', 'Fear', 'Happy', 'Sad', 'Surprise', and 'Neutral' facial expressions respectively. From the table, we observe that the 'Happy' class performs the best and achieves accuracy of nearly 97%. On the other hand, 'Sad' classifier performs the worst and achieves accuracy of only about 44%, where most of the 'Sad' class images are mis-classified as 'Neutral' class. We find that this bad performance is caused by the small number of training samples in 'Sad' class. Note that the 'Fear' class also has a small number of training samples and performs second worst. In order to prove our speculation, we run the same experiment with KDEF Dataset, in which it has same number of images in all classes. **Table 13** presents the confusion matrix of HOG descriptor with linear SVM classifier applied to the CK+ Dataset. From the table, we observe that 'Sad' classifier performs much better with the KDEF Dataset, proving that our speculation is correct. In Section 11, we will present more experiment results on KDEF, MUG, and JAFFE Datasets.

Table 11 Test accuracies of PCA+LDA, SVM, and AdaBoost (CK+ Dataset) with BRIEF and Gaussian-BRIEF descriptors.

	PCA+LDA	SVM	AdaBoost
BRIEF	**83.7 ± 3.4**	83.2 ± 3.4	**79.7 ± 4.4**
G-BRIEF	81.7 ± 5.7	**83.9 ± 4.7**	78.3 ± 6.1

Table 12 Confusion matrix of HOG descriptor with linear SVM classifier (CK+ Dataset).

%	An	Di	Fe	Ha	Sa	Su	Ne
An	**76.4**	6.8	1.1	1.1	1.1	2.3	11.2
Di	1.9	**90.4**	1.0	1.9	0.0	1.9	2.9
Fe	3.6	1.8	**71.3**	3.6	1.8	3.6	14.3
Ha	0.0	0.0	0.0	**97.1**	0.0	0.7	2.2
Sa	10.2	6.8	5.1	8.5	**44.0**	0.0	25.4
Su	0.0	0.0	1.2	0.0	0.6	**93.4**	4.8
Ne	0.7	0.0	0.6	0.5	0.7	0.2	**97.3**

9. Feature Fusion Investigation

Since all feature descriptors possess different characteristics, one common question would be—could the test accuracies be further improved by using multiple feature descriptors for image classification? We have analyzed this issue by considering the following experiment. We combined the LBP, HOG, and BRIEF descriptors and tested the outcomes with AdaBoost classifier. It would be also interesting to realize this experiment by using SVM or other techniques such as multiple kernel learning (MKL) [66], [67]. Similar to AdaBoost, MKL has the ability to perform feature selection and classification simultaneously. However, our main objective is to analyze the description power of the feature descriptors by explicitly counting the number of feature descriptors selected by AdaBoost. To this end, we chose to consider MKL as our future work for FER.

Table 14 summarizes our experiment results. From the table, we can observe that the combined LBP & HOG & BRIEF descriptors perform the best, followed by HOG descriptor, LBP descriptor, and BRIEF descriptor. In addition, **Table 15** shows the number of feature descriptors selected by AdaBoost during the combined-features experiment. We observe that HOG descriptor almost always have the largest number of selection, indicating that HOG descriptor has more description power than other feature descriptors in the AdaBoost classification experiment.

10. Image Pre-processing Investigation

In this section, we investigate the effects of image pre-processing towards the FER test performances. Specifically, we investigated the effect of image resolutions towards the test performances. In addition to the effect of image resolutions, we also compared the results under normal pre-processing, i.e., face detection and cropping procedures described in Section 3, to the results with additional pre-processing steps. We considered his-

Table 13 Confusion matrix of HOG descriptor with linear SVM classifier (KDEF Dataset).

%	An	Di	Fe	Ha	Sa	Su	Ne
An	**78.7**	7.1	6.4	1.4	1.1	2.5	2.8
Di	4.1	**80.7**	4.1	2.2	2.2	4.5	2.2
Fe	5.0	4.0	**65.6**	4.3	5.7	4.3	11.1
Ha	0.7	1.0	0.4	**95.2**	0.7	1.0	1.0
Sa	1.1	0.4	0.4	1.1	**93.8**	1.4	1.8
Su	3.1	5.8	6.2	2.7	5.0	**76.0**	1.2
Ne	0.3	1.0	6.3	0.7	3.3	1.6	**86.8**

Table 14 Test accuracies of AdaBoost classifier with combined features in CK+ Dataset.

Descriptors	AdaBoost
LBP	81.2 ± 3.9
HOG	85.7 ± 3.0
BRIEF	79.7 ± 4.4
LBP+HOG+BRIEF	**87.3 ± 3.8**

Table 15 Percentages of feature selected by AdaBoost in 20 validation cycles with CK+ Dataset.

Descriptors	An (%)	Di (%)	Fe (%)	Ha (%)	Sa (%)	Su (%)	Ne (%)
LBP	30	24	**35**	29	24	28	33
HOG	**40**	**54**	**35**	35	**41**	**42**	**42**
BRIEF	30	22	30	**36**	35	30	25

Table 16 Comparison of size of feature vectors at different image resolution.

Descriptors	$12 \times 12 = 144$	$24 \times 24 = 576$	$48 \times 48 = 2{,}304$	$96 \times 96 = 9{,}216$
LBP	$3^2 \times 59 = 531$	$8^2 \times 59 = 3{,}776$	$13^2 \times 59 = 9{,}971$	$19^2 \times 59 = 21{,}299$
HOG	$(1^2 + 2^2) \times 36 = 180$	$(2^2 + 5^2) \times 36 = 1{,}044$	$(5^2 + 11^2) \times 36 = 5{,}256$	$(5^2 + 11^2 + 23^2) \times 36 = 24{,}300$
BRIEF	2,500	5,000	10,000	20,000

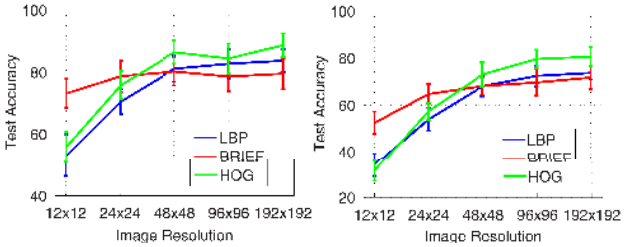

Fig. 15 Test accuracies of AdaBoost classifier under different image resolution settings in CK+ Dataset (left) and KDEF Dataset (right).

togram equalization (HE) and median filtering (MF) in our experiments. HE could potentially enhances facial features that are not obvious in the low contrast images. On the other hand, MF could filter the image noises and smoothen the facial images. Note that we do not consider illumination normalization since the datasets we are using now are collected under controlled settings (i.e., have same illumination within datasets).

10.1 Effect of Image Resolution

We investigate the effect of image resolution on test accuracies in this section. Looking for an optimal image resolution that produces high test accuracy and low computational cost is important but is not commonly focused in research analysis. **Table 16** summarizes the size of all the five feature descriptors used in our experiments under different image resolution settings. The fourth column (48×48 = 2,304) is the original setting in our previous experiments. Specifically, LBP operator has size of 12×12 pixels and shift horizontally and vertically by 3 pixels, resulting in a feature vector with size of 13×13×59 = 9,971. HOG operator has two combined settings—block size of 16×16 pixels and 8×8 pixels respectively, resulting in a feature vector with size of (5×5 + 11×11)×36 = 5,256. BRIEF descriptor was set to have 10,000 values, which is close to the LBP descriptor size of 9,971. We only considered LBP, HOG, and BRIEF descriptors in this section. The dimensions of Gabor and Haar descriptors of images at higher resolution are too large and significant amount of computational resources are required for the experiments.

Figure 15 summarizes the test accuracies of CK+ and KDEF Datasets under different image resolutions with AdaBoost classifier. Test results of both datasets are consistent. We observe that LBP and HOG descriptors produce test accuracies higher than BRIEF descriptor at high resolution images (≥ 48×48 pixels). In contrast, BRIEF descriptor produces higher test accuracies than LBP and HOG descriptors at low resolution images (< 48×48 pixels). Furthermore, BRIEF descriptor can produce surprisingly good test accuracies across all different image resolutions despite its much lower computational cost. We also observe that HOG descriptor almost always performs better than LBP descriptor across all image resolution settings. In practice, we recommend to normalize a high resolution image (≥ 48×48) to size of 48×48 pixels (because higher image resolution would produce similar performance) and use HOG descriptor for FER. When original image resolution is lower than 48×48 pixels, we recommend to use BRIEF descriptor.

10.2 Effect of Image Filtering

We find that comparing the results of HE and MF with the results of normal processing is challenging because their results strongly depend on the feature descriptors and classifiers. For the ease of understanding, we summarized the test results in five tables (**Tables 21**, **22**, **23**, **24** and **25**), in which each of them corresponds to Gabor, Haar, LBP, HOG, and BRIEF descriptors. For Gabor descriptor, we find that HE always produces the best test results, while MF always produces the worse test results in all three classifiers. This is reasonable as HE could enhances facial features that are not obvious in the low contrast images while MF smoothen the facial features and would deteriorate the test results.

For HOG descriptor, we find that all three classifiers produce the best results without additional pre-processing. Different to Gabor filter, this indicates that HOG descriptor could capture the edge features in the facial images efficiently even without additional HE. This again would lead us to favor HOG descriptor in FER. For Haar, LBP, and BRIEF descriptors, the test performances vary depending on the chosen classifiers and we could not make a clear conclusion.

11. Generalization Tests

We performed three types of generalization test in this section. First, we applied the same experiments to the MUG, KDEF, and JAFFE Datasets. Second, we combined all the CK+, MUG, KDEF, and JAFFE Datasets into a large dataset (totally 2,748 images), performed the same procedures, and report its test performance. Third, we trained the PCA+LDA, SVM, and AdaBoost classifiers with all CK+ Dataset images and tested classifiers' performance with all MUG, KDEF, and JAFFE Datasets.

11.1 MUG, KDEF, and JAFFE Datasets

Table 17 summarizes the test accuracies of five feature descriptors with RFS+LDA and PCA+LDA classifiers applied to the MUG, KDEF, and JAFFE Datasets. Again, as expected, PCA performs better than RFS because PCA reduces the size of feature descriptors while capturing the most important information systematically. We also find that HOG descriptor produces the best test accuracies in PCA+LDA classifier.

Table 18 summarizes the test accuracies of five feature descriptors with SVM and AdaBoost classifiers applied to the MUG, KDEF, and JAFFE Datasets. We observe that SVM classifier performs better than AdaBoost classifier across all the three datasets and HOG descriptor produces the best test accuracies in SVM classifier. Moreover, we find that SVM classifier performs better

Table 17 Test accuracies of Random Feature Selection+LDA and PCA+LDA.

Descriptors	MUG Dataset		KDEF Dataset		JAFFE Dataset	
	RFS+LDA	PCA+LDA	RFS+LDA	PCA+LDA	RFS+LDA	PCA+LDA
Gabor	**76.7 ± 4.3**	80.3 ± 5.0	69.2 ± 4.3	71.5 ± 3.4	62.9 ± 11.3	79.1 ± 8.2
Haar	75.4 ± 5.1	79.4 ± 5.1	68.8 ± 4.6	70.2 ± 3.5	**65.0 ± 11.2**	76.0 ± 10.9
LBP	56.7 ± 5.1	77.8 ± 3.7	52.4 ± 4.3	72.0 ± 4.3	34.3 ± 9.4	63.3 ± 6.2
HOG	71.7 ± 4.8	**82.6 ± 4.5**	**68.0 ± 4.7**	**77.1 ± 3.7**	54.3 ± 11.2	**85.5 ± 6.6**
BRIEF	54.1 ± 6.6	74.5 ± 3.6	50.6 ± 4.7	69.9 ± 3.8	37.4 ± 13.3	69.3 ± 8.2

Table 18 Test accuracies of SVM and AdaBoost.

Descriptors	MUG Dataset		KDEF Dataset		JAFFE Dataset	
	SVM	AdaBoost	SVM	AdaBoost	SVM	AdaBoost
Gabor	82.7 ± 3.4	**77.5 ± 4.1**	72.6 ± 5.5	71.3 ± 3.1	82.4 ± 7.1	**69.0 ± 10.3**
Haar	78.6 ± 4.7	73.5 ± 4.6	69.0 ± 6.4	67.6 ± 5.0	77.1 ± 10.2	67.4 ± 12.5
LBP	79.0 ± 4.2	67.9 ± 5.6	74.7 ± 5.2	66.7 ± 3.9	55.7 ± 9.8	49.5 ± 9.6
HOG	**85.3 ± 4.2**	77.0 ± 5.3	**80.2 ± 4.1**	**75.2 ± 4.0**	**89.5 ± 6.3**	64.0 ± 11.0
BRIEF	81.4 ± 4.2	71.7 ± 4.4	73.4 ± 5.7	68.8 ± 5.4	72.6 ± 10.4	62.1 ± 13.0

Table 19 Test accuracies of Random Feature Selection+LDA and PCA+LDA.

Descriptors	Combined Dataset		CK+ Dataset	
	RFS+LDA	PCA+LDA	RFS+LDA	PCA+LDA
Gabor	55.2 ± 2.7	56.6 ± 2.4	79.4 ± 5.2	82.7 ± 3.4
Haar	55.4 ± 2.8	58.6 ± 3.4	80.8 ± 4.9	83.4 ± 4.2
LBP	47.0 ± 3.5	63.7 ± 2.1	66.4 ± 6.0	86.4 ± 2.6
HOG	**60.5 ± 3.5**	**68.1 ± 3.2**	**82.5 ± 4.1**	**90.9 ± 3.2**
BRIEF	44.0 ± 3.2	59.3 ± 3.5	67.2 ± 5.9	83.7 ± 3.4

Table 20 Test accuracies of SVM and AdaBoost.

Descriptors	Combined Dataset		CK+ Dataset	
	SVM	AdaBoost	SVM	AdaBoost
Gabor	61.6 ± 2.4	62.8 ± 3.1	83.6 ± 3.4	81.1 ± 5.8
Haar	59.9 ± 2.8	60.0 ± 3.6	80.2 ± 3.5	78.0 ± 4.1
LBP	70.5 ± 2.7	59.6 ± 3.0	86.0 ± 4.0	81.2 ± 3.9
HOG	**73.3 ± 3.3**	**63.2 ± 3.2**	**91.2 ± 3.2**	**85.7 ± 3.0**
BRIEF	73.0 ± 2.7	59.3 ± 2.5	83.2 ± 3.4	79.7 ± 4.4

than the PCA+LDA classifier in Table 17.

11.2 Combined Datasets

Table 19 summarizes the generalization performance of RFS+LDA and PCA+LDA classifiers in the combined dataset (we also repeat the test results of CK+ Dataset for direct comparison purpose). While the combined dataset is about 4 times larger than CK+ Dataset, we observe that both RFS+LDA and PCA+LDA classifiers perform worse in the combined dataset. This is not surprising, as the four datasets are collected under controlled indoor environment with different backgrounds and light settings. Moreover, the four datasets have different demographical settings, e.g., CK+ Dataset was collected in North America, MUG and KDEF Datasets were collected in Europe, while JAFFE Dataset was collected in Asia. Our generalization test performance conforms with previous FER studies, suggesting that we need to collect more FER images in the wild in order to achieve a more robust FER [11], [21].

Table 20 summarizes the generalization performance of SVM and AdaBoost classifiers in the combined dataset (we repeat the test results of CK+ Dataset for direct comparison purpose). Similar to the preceding discussion, we observe that both SVM and AdaBoost classifiers perform worse in the case of combined dataset, suggesting that we need to collect more FER images in the wild in order to achieve a more robust FER.

11.3 Cross Datasets

In addition to the combined dataset, we also trained the PCA+LDA, SVM, and AdaBoost classifiers with all CK+ images and tested classifiers' performance with all MUG, KDEF, and JAFFE images. We summarized the test accuracies of the three datasets in **Table 26**. From the table, we can observe that in general, all test performances are much worse than the performances of individual CK+ Dataset and combined datasets cases due to the reasons discussed in Section 11.2. By referring to the dataset images (Fig. 1–3), we can observe that each dataset was collected under very different controlled indoor environment, which eventually lead to these unsatisfactory results. Similar to the previous sections, these test results suggest us to collect more FER images in the wild in order to achieve more robust FER.

In addition, we also observed that JAFFE Dataset produces much worse cross-dataset results than MUG and KDEF Datasets. Upon careful investigation, we found that JAFFE Database has a few ambiguous facial expressions posed by models. **Figure 16** illustrates a few faces with ambiguous facial expression. For instance, the first image has a label of 'Angry' but may be poten-

Table 21 Test accuracies of Gabor descriptor (CK+ Dataset) by using PCA+LDA, SVM, and AdaBoost classifiers with normal, histogram equalization, median filtering pre-processing.

	PCA+LDA	SVM	AdaBoost
Normal	82.7 ± 3.4	83.6 ± 3.4	81.1 ± 5.8
HE	**84.4 ± 3.4**	**84.7 ± 4.8**	**82.5 ± 4.7**
MF	80.0 ± 5.3	80.1 ± 4.5	78.8 ± 4.8

Table 22 Test accuracies of Haar descriptor (CK+ Dataset) by using PCA+LDA, SVM, and AdaBoost classifiers with normal, histogram equalization, median filtering pre-processing.

	PCA+LDA	SVM	AdaBoost
Normal	83.4 ± 4.2	80.2 ± 3.5	**78.0 ± 4.1**
HE	**84.5 ± 4.3**	**81.1 ± 6.1**	77.1 ± 6.1
MF	82.7 ± 4.6	77.7 ± 5.6	74.5 ± 5.6

Table 23 Test accuracies of LBP descriptor (CK+ Dataset) by using PCA+LDA, SVM, and AdaBoost classifiers with normal, histogram equalization, median filtering pre-processing.

	PCA+LDA	SVM	AdaBoost
Normal	86.4 ± 2.6	**86.0 ± 4.0**	81.2 ± 3.9
HE	87.1 ± 5.3	84.6 ± 4.9	**81.3 ± 5.0**
MF	**88.0 ± 4.7**	84.4 ± 5.4	75.6 ± 6.7

Table 24 Test accuracies of HOG descriptor (CK+ Dataset) by using PCA+LDA, SVM, and AdaBoost classifiers with normal, histogram equalization, median filtering pre-processing.

	PCA+LDA	SVM	AdaBoost
Normal	**90.9 ± 3.2**	**91.2 ± 3.2**	**85.7 ± 3.0**
HE	88.9 ± 5.5	88.6 ± 4.0	82.0 ± 5.3
MF	88.3 ± 4.0	87.1 ± 6.0	83.1 ± 4.8

Table 25 Test accuracies of BRIEF descriptor (CK+ Dataset) by using PCA+LDA, SVM, and AdaBoost classifiers with normal, histogram equalization, median filtering pre-processing.

	PCA+LDA	SVM	AdaBoost
Normal	**83.7 ± 3.4**	83.2 ± 3.4	79.7 ± 4.4
HE	82.8 ± 5.0	**83.5 ± 4.5**	79.4 ± 3.8
MF	83.0 ± 5.5	83.0 ± 5.3	**80.6 ± 5.7**

Table 26 Test accuracies of MUG, KDEF, and JAFFE Datasets (1st, 2nd, 3rd numbers in all the triplets) with PCA+LDA, SVM, and AdaBoost classifiers by using CK+ Dataset as training data.

	PCA+LDA	SVM	AdaBoost
Gabor	24.6 / 24.0 / 11.7	29.6 / 24.9 / 15.5	26.2 / 22.9 / 16.0
Haar	27.6 / 27.0 / 15.0	31.2 / 32.2 / 16.9	33.3 / 31.3 / 16.0
LBP	26.9 / 30.2 / 14.1	28.1 / 27.6 / 13.6	26.4 / 26.6 / 20.7
HOG	31.9 / 35.3 / 13.6	29.5 / 35.5 / 18.8	27.8 / 33.3 / 23.0
BRIEF	27.5 / 26.3 / 11.7	29.4 / 25.1 / 15.0	29.1 / 26.1 / 17.4

Fig. 16 Ambiguous facial expression labels in JAFFE Dataset.

tially perceived as 'Sad' expression. Since JAFFE Database is small, these ambiguity represents about 5% of the all 213 images. We believe that this is the main reason that JAFFE Dataset performs the worst in our cross-dataset experiments.

12. Computational Efficiency

Feature descriptor should be computationally efficient while leading to the best classification results. In this section, we analyze the computational cost of five feature descriptors in term of number of multiplication (NOM) and summation (NOS) operations. We first list up a few assumptions and describe our calculation process. In all five feature descriptors, we consider a grayscale image with size of 48×48 pixels.

Among the five feature descriptors, Gabor descriptor has the most expensive computational cost because it involves 48×48 summation and 48×48 multiplication operations. On the other hand, taking the computational cost of integral image into consideration, Haar descriptor has computational cost of only

$$NOS_{Haar} = 15 \times numberOfSelectedFeatures + imageSize^2 \times 3 \,. \quad (12)$$

The computation of LBP descriptor involves thresholding, binary mapping, and histogram binning operations. For simplicity, we assume one binary thresholding operation is equivalent to one summation operation. Besides, we assume that binary mapping is realized with Lookup Table (LUT) technique and is considered as one summation operation as well. We also assume that histogram binning operation is equivalent to one summation operation. Under these assumptions, our implementation of *one* LBP descriptor has computational cost of

$$NOS_{LBP} = \left(\frac{imageSize}{4}\right)^2 \times (8 + 1 + 1) \,. \quad (13)$$

The computation of HOG descriptor involves convolution, square-root, arctangent, histogram binning, and normalization operations. However, it has surprisingly low computational cost when LUT technique is employed. Convolution process (by applying 1-D horizontal and vertical Sobel masks) seems to be expensive but it is in fact equivalent to one summation operation for each gradient map computation process. Square-root and arctangent operations can also be replaced with LUT technique since there are only 511×511 possibilities for an 8-bit grayscale image. Hence, we count these as two summation operations for every pixel when calculating gradient magnitude and gradient orientation maps. Subsequent histogram binning operations is considered equivalent to one summation operation. Local normalization involves 35 summation and 72 multiplication operations for each block. Under these assumptions, our implementation of *one* HOG descriptor has computational cost of

$$NOS_{HOG} = \left(\left(\frac{imageSize}{4}\right)^2 \times (2 + 2 + 1) + 35\right) \,,$$
$$NOM_{HOG} = 72 \,. \quad (14)$$

BRIEF descriptor is well-known for its simplicity and fast computation. It involves only five comparison operations (equivalent to five summation operations) and one binary mapping operation (counted as one summation operations). BRIEF descriptor is also the only feature that has computation cost independent of image size. Overall, our implementation of *one* BRIEF descriptor has computational cost of

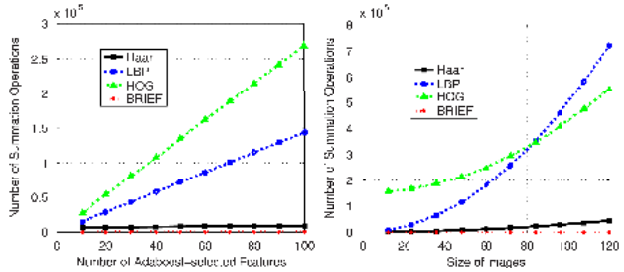

Fig. 17 Number of summation operations of feature descriptors with varied number of features (top) and varied size of images (bottom).

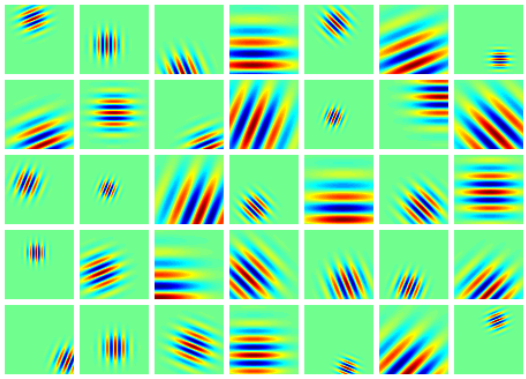

Fig. 18 Visualization of Gabor descriptors selected by AdaBoost classifier. Each column represents the first five Gabor descriptors selected by 'Angry', 'Disgust', 'Fear', 'Happy', 'Sad', 'Surprise', 'Neutral' classifier respectively.

$$NOS_{BRIEF} = (5+1)\,. \tag{15}$$

In descending order, Gabor descriptor has the most expensive computational cost, followed by HOG, LBP, Haar, and BRIEF descriptors. Assuming that one multiplication operation is equivalent to ten summation operations, we show two interesting plots in **Fig. 17**. We investigated the number of summation operations by varying the number of selected features and changing the image size. We observe that all features' computational cost increase approximately linearly when number of features increase. On the other hand, LBP and HOG descriptors have exponentially increasing summation operations when image size increases. Nevertheless, all features have very low computation cost in modern computers. For instance, it takes less than 15 μs to compute one LBP and HOG descriptor respectively in MATLAB (C & C++ implementations). It is worth noting that Haar and BRIEF descriptors have even lower computational cost in the two cases shown in Fig. 17. Haar descriptor has a low computational cost thanks to the integral image technique while BRIEF descriptor has a computational cost independent of the image size.

13. Feature Visualization

Figures 18 and **19** visualize Gabor and Haar descriptors selected by AdaBoost classifiers in CK+ Dataset. These figures show some insights about the size and position of the descriptors selected by the AdaBoost classifier. We can see that most Gabor and Haar descriptors are small and concentrate at the image center. In **Fig. 20**, we overlap all 80 feature descriptors selected by AdaBoost classifier. We can observe that all feature descriptors concentrate at the image center. The observation also suggests us

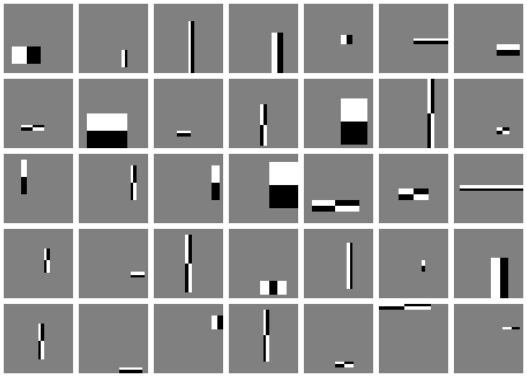

Fig. 19 Visualization of Haar descriptors selected by AdaBoost classifier. Each column represents the first five Haar descriptors selected by 'Angry', 'Disgust', 'Fear', 'Happy', 'Sad', 'Surprise', 'Neutral' classifier respectively.

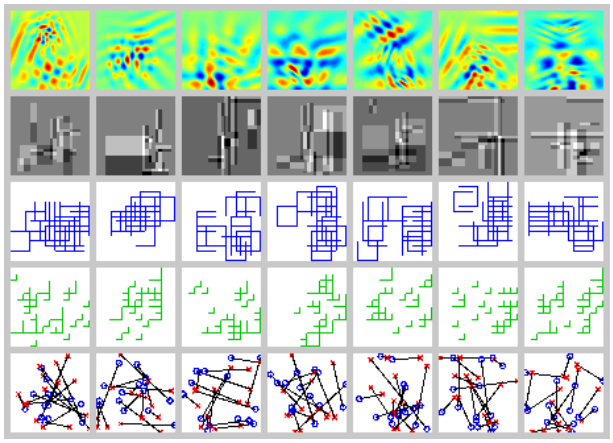

Fig. 20 Visualization of 80 overlapping Gabor, Haar, LBP, HOG, and BRIEF descriptors (row-wise) of 'Angry', 'Disgust', 'Fear', 'Happy', 'Sad', 'Surprise', and 'Neutral' classifier (column-wise) respectively.

to put more weights on these location when we extract the feature descriptors from the face images. This concept is similar to the idea in Ref. [11], when LBP descriptor is extracted with different weighting effect based on their extraction location.

14. Conclusion

In this paper, we empirically evaluate *five* feature descriptors, namely Gabor, Haar, LBP, HOG, and BRIEF descriptors in FER. We examine each feature descriptor by considering *six* classification methods, such as k-NN, LDA, SVM, and AdaBoost with *four* unique facial expression datasets. In the end, we identified HOG descriptor as the best feature descriptor for FER when image resolution of a detected face is higher than 48×48. On the other hand, when the image resolution of the detected face is smaller than 48×48, our experiment results show that BRIEF descriptor performs the best. In general, Gabor descriptor performs well but has a higher computational cost. In addition to the test accuracies, we presented confusion matrices of FER. We analyzed the effect of combined features and image resolutions on FER performance. We also generalized our experiments to other datasets, analyzed the computational efficiency of each feature descriptors, and visualized the feature descriptors selected by AdaBoost classifier.

In this study, we only consider frontal facial expression images.

The use of facial expression dataset under different head poses is necessary to consolidate our findings. We also focus FER on single image. Temporal information provides strong clues about facial expression and should be carefully considered in the future.

Acknowledgments This research was partially supported by Japanese Government (MEXT) Scholarship Program and JSPS Grants-in-Aid for Scientific Research Program. The author also would like to thank the reviewers for their valuable insights, comments, and suggestions.

References

[1] Gabor, D.: Theory of Communication, *Journal of the Institution of Electrical Engineers*, Vol.93, No.26, pp.429–457 (1946).
[2] Papageorgiou, C., Oren, M. and Poggio, T.: A General Framework for Object Detection, *Proc. 6th International Conference on Computer Vision*, pp.555–562 (1998).
[3] Ojala, T., Pietikainen, M. and Harwood, D.: Performance Evaluation of Texture Measures with Classification based on Kullback Discrimination of Distributions, *Proc. 12th International Conference on Pattern Recognition*, pp.582–585 (1994).
[4] Dalal, N. and Triggs, B.: Histograms of Oriented Gradients for Human Detection, *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.886–893 (2005).
[5] Ozuysal, M., Calonder, M., Lepetit, V. and Fua, P.: Fast Keypoint Recognition Using Random Ferns, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.32, No.3, pp.448–461 (2010).
[6] Liew, C.F. and Yairi, T.: A Comparison Study of Feature Spaces and Classification Methods in Facial Expression Recognition, *Proc. IEEE International Conference on Robotics and Biomimetics*, pp.1294–1299 (2013).
[7] Rudovic, O., Patras, I. and Pantic, M.: Regression-based Multi-View Facial Expression Recognition, *Proc. 20th International Conference on Pattern Recognition*, pp.4121–4124 (2010).
[8] Pantic, M. and Rothkrantz, L.: Expert System for Automatic Analysis of Facial Expressions, *Journal of Image and Vision Computing*, Vol.18, No.11, pp.881–905 (2000).
[9] Asthana, A., Saragih, J., Wagner, M. and Goecke, R.: Evaluating AAM Fitting Methods for Facial Expression Recognition, *Proc. 3rd International Conference on Affective Computing and Intelligent Interaction Workshops*, pp.1–8 (2009).
[10] Li, Y., Wang, S., Zhao, Y. and Ji, Q.: Simultaneous Facial Feature Tracking and Facial Expression Recognition, *IEEE Trans. Image Processing*, Vol.22, No.7, pp.2559–2573 (2013).
[11] Shan, C., Gong, S. and McOwan, P.: Facial Expression Recognition based on Local Binary Patterns: A Comprehensive Study, *Journal of Image and Vision Computing*, Vol.27, No.6, pp.803–816 (2009).
[12] Zhang, Z., Lyons, M., Schuster, M. and Akamatsu, S.: Comparison between Geometry-based and Gabor-wavelets-based Facial Expression Recognition using Multi-layer Perceptron, *Proc. 3rd IEEE International Conference on Automatic Face and Gesture Recognition*, pp.454–459 (1998).
[13] Gonzalez, I., Sahli, H., Enescu, V. and Verhelst, W.: Context-independent Facial Action Unit Recognition Using Shape and Gabor Phase Information, *Proc. 4th International Conference on Affective Computing and Intelligent Interaction*, pp.548–557 (2011).
[14] Tian, Y., Kanade, T. and Cohn, J.: Recognizing Action Units for Facial Expression Analysis, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.23, No.2, pp.97–115 (2001).
[15] Sadeghi, H., Raie, A. and Mohammadi, M.: Facial Expression Recognition Using Geometric Normalization and Appearance Representation, *Proc. 8th Iranian Conference on Machine Vision and Image Processing*, pp.159–163 (2013).
[16] Happy, S.L. and Routray, A.: Automatic Facial Expression Recognition Using Features of Salient Facial Patches, *IEEE Trans. Affective Computing*, Vol.6, No.1, pp.1–12 (2015).
[17] Zheng, W.: Multi-View Facial Expression Recognition Based on Group Sparse Reduced-Rank Regression, *IEEE Trans. Affective Computing*, Vol.5, No.1, pp.71–85 (2014).
[18] Eleftheriadis, S., Rudovic, O. and Pantic, M.: Discriminative Shared Gaussian Processes for Multiview and View-Invariant Facial Expression Recognition, *IEEE Trans. Image Processing*, Vol.24, No.1, pp.189–204 (2015).
[19] Lyons, M. and Akamatsu, S.: Coding Facial Expressions with Gabor Wavelets, *Proc. 3rd IEEE International Conference on Automatic Face and Gesture Recognition*, pp.200–205 (1998).
[20] Wu, T., Bartlett, M. and Movellan, J.: Facial Expression Recognition Using Gabor Motion Energy Filters, *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp.42–47 (2010).
[21] Bartlett, M., Littlewort, G., Frank, M., Lainscsek, C., Fasel, I. and Movellan, J.: Automatic Recognition of Facial Actions in Spontaneous Expressions, *Journal of Multimedia*, Vol.1, No.6, pp.22–35 (2006).
[22] Bartlett, M., Littlewort, G., Lainscsek, C., Fasel, I., Frank, M. and Movellan, J.: Fully Automatic Facial Action Recognition in Spontaneous Behavior, *Proc. 7th International Conference on Automatic Face and Gesture Recognition*, pp.223–228 (2006).
[23] Littlewort, G., Bartlett, M., Fasel, I., Susskind, J. and Movellan, J.: Dynamics of Facial Expression Extracted Automatically from Video, *Journal of Image and Vision Computing*, Vol.24, No.6, pp.615–625 (2006).
[24] Lee, C. and Shih, C.: Gabor Feature Selection for Facial Expression Recognition, *Proc. International Conference on Signals and Electronic Systems*, pp.139–142 (2010).
[25] Xu, C., Dong, C., Feng, Z. and Cao, T.: Facial Expression Pervasive Analysis Based on Haar-Like Features and SVM, *Proc. The International Conference on E-business Technology and Strategy*, pp.521–529 (2012).
[26] Jung, S., Kim, D., An, K. and Chung, M.: Efficient Rectangle Feature Extraction for Real-time Facial Expression Recognition based on AdaBoost, *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.1941–1946 (2005).
[27] Nagi, G., Rahmat, R., Khalid, F. and Taufik, M.: Region-Based Facial Expression Recognition in Still Images, *Journal of Information Processing Systems*, Vol.9, No.1, pp.173–188 (2013).
[28] Dahmane, M. and Meunier, J.: Emotion Recognition using Dynamic Grid-based HoG Features, *Proc. IEEE International Conference on Automatic Face and Gesture Recognition Workshops*, pp.884–888 (2011).
[29] Orrite, C., Ganan, A. and Rogez, G.: HOG-Based Decision Tree for Facial Expression Classification, *Proc. 4th Iberian Conference on Pattern Recognition and Image Analysis*, pp.176–183 (2009).
[30] Chen, S., Tian, Y., Liu, Q. and Metaxas, D.: Recognizing Expressions from Face and Body Gesture by Temporal Normalized Motion and Appearance Features, *Journal of Image and Vision Computing*, Vol.31, No.2, pp.175–185 (2013).
[31] Bosch, A., Zisserman, A. and Munoz, X.: Image Classifcation Using Random Forests and Ferns, *Proc. 11th IEEE International Conference on Computer Vision*, pp.1–8 (2007).
[32] Moeini, A., Moeini, H. and Faez, K.: Pose-Invariant Facial Expression Recognition Based on 3D Face Reconstruction and Synthesis from a Single 2D Image, *Proc. 22nd International Conference on Pattern Recognition*, pp.1746–1751 (2014).
[33] Vapnik, V. and Lerner, A.: Pattern Recognition Using Generalized Portrait Method, *Journal of Automation and Remote Control*, Vol.24 (1963).
[34] Cortes, C. and Vapnik, V.: Support Vector Networks, *Journal of Machine Learning*, Vol.20, pp.273–297 (1995).
[35] Freund, Y. and Schapire, R.: A Decision-theoretic Generalization of On-line Learning and an Application to Boosting, *Journal of Computer and System Sciences*, Vol.55, No.1, pp.119–139 (1997).
[36] Kanade, T., Cohn, J. and Tian, Y.: Comprehensive Database for Facial Expression Analysis, *Proc. 4th International Conference on Automatic Face and Gesture Recognition*, pp.46–53 (2000).
[37] Lucey, P., Cohn, J., Kanade, T., Saragih, J., Ambadar, Z. and Matthews, I.: The Extended Cohn-Kanade Dataset (CK+): A Complete Dataset for Action Unit and Emotion-specified Expression, *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp.94–101 (2010).
[38] Aifanti, N., Papachristou, C. and Delopoulos, A.: The MUG Facial Expression Database, *Proc. 11th International Workshop on Image Analysis for Multimedia Interactive Services*, pp.1–4 (2010).
[39] Lundqvist, D., Flykt, A. and Ohman, A.: The Karolinska Directed Emotional Faces - KDEF, *CD ROM from Department of Clinical Neuroscience, Psychology section, Karolinska Institutet*.
[40] Lyons, M., Budynek, J. and Akamatsu, S.: Automatic Classification of Single Facial Images, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.21, No.12, pp.1357–1362 (1999).
[41] MATLAB R2013a Documentation - Face Detection and Tracking, available from ⟨http://www.mathworks.com/help/vision/examples/face-detection-and-tracking.html⟩ (accessed 2013-07-22).
[42] Ekman, P.: An Argument for Basic Emotions, *Cognition and Emotion*, Vol.6, pp.169–200 (1992).
[43] Daugman, J.: Uncertainty Relation for Resolution in Space, Spatial Frequency, and Orientation Optimized by Two-dimensional Visual Cortical Filters, *Journal of the Optical Society of America*, Vol.2, No.7, pp.1160–1169 (1985).

[44] Daugman, J.: How Iris Recognition Works, *Journal of the Optical Society of America*, Vol.14, No.1, pp.21–30 (2004).
[45] Jain, A., Ross, A. and Prabhakar, S.: Fingerprint Matching using Minutiae and Texture Features, *Proc. International Conference on Image Processing*, pp.282–285 (2001).
[46] Viola, P. and Jones, M.: Rapid Object Detection Using a Boosted Cascade of Simple Features, *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.511–518 (2001).
[47] Ahonen, T., Hadid, A. and Pietikanen, M.: Face Description with Local Binary Patterns: Application to Face Recognition, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.28, No.12, pp.2037–2041 (2006).
[48] Online resource, available from ⟨http://www.flickr.com/photos/58842866@N08/5388738458/sizes/o/in/photostream/⟩ under Attribution Creative Commons License on July 25, 2013.
[49] Lowe, D.: Object Recognition from Local Scale-invariant Features, *Proc. 7th International Conference on Computer Vision*, pp.1150–1157 (1999).
[50] Online resource, available from ⟨http://www.flickr.com/photos/davedehetre/5328057917/sizes/o/in/photostream/⟩ under Attribution Creative Commons License on July 25, 2013.
[51] Ozuysal, M., Calonder, M., Lepetit, V. and Fua, P.: Keypoint Recognition Using Random Trees, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.28, No.9, pp.1465–1479 (2006).
[52] Calonder, M., Lepetit, V., Ozuysal, M., Trzcinski, T., Strecha, C. and Fua, P.: BRIEF: Computing a Local Binary Descriptor Very Fast, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.34, No.7, pp.1281–1298 (2012).
[53] Online resource, available from ⟨http://www.fotopedia.com/items/flickr-3714021304/⟩ under Attribution and ShareAlike Creative Commons License on July 25, 2013.
[54] Murphy, K.: *Machine Learning: A Probabilistic Perspective Chapter 4*, The MIT Press (2012).
[55] MATLAB R2013b Documentation — Discriminant Analysis, available from ⟨http://www.mathworks.com/help/stats/discriminant-analysis.html⟩ (accessed 2014-01-01).
[56] Wikipedia — Linear Discriminant Analysis, available from ⟨http://en.wikipedia.org/wiki/Linear_discriminant_analysis⟩ (accessed 2014-01-01).
[57] Lindsay, S.: A Tutorial on Principal Component Analysis (2002), available from ⟨http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf⟩ (accessed 2014-03-26).
[58] Halko, N., Martinsson, P., Shkolnisky, Y. and Tygert, M.: An Algorithm for the Principal Component Analysis of Large Data Sets, *SIAM Journal on Scientific Computing*, Vol.33, No.5, pp.2580–2594 (2011).
[59] Meyer, D. and Wien, T.: Support Vector Machines - The Interface to libsvm in package e1071 (2012).
[60] Burges, J.: A Tutorial on Support Vector Machines for Pattern Recognition, *Journal of Data Mining and Knowledge Discovery*, Vol.2, No.2, pp.121–167 (1998).
[61] MATLAB R2013a Documentation - Support Vector Machines, available from ⟨http://www.mathworks.com/help/stats/support-vector-machines.html⟩ (accessed 2013-07-16).
[62] Platt, J.: Fast Training of Support Vector Machines Using Sequential Minimal Optimization, *Advances in Kernel Methods*, Schölkopf, B., Burges, C.J.C. and Smola, A.J. (Eds.), pp.185–208, MIT Press Cambridge (1999).
[63] Freund, Y. and Schapire, R.: A Decision-theoretic Generalization of On-line Learning and an Application to Boosting, *Proc. 2nd European Conference on Computational Learning Theory*, pp.23–37 (1995).
[64] Ahonen, T., Matas, J., He, C. and Pietikäinen, M.: Rotation Invariant Image Description with Local Binary Pattern Histogram Fourier Features, *Proc. 16th Scandinavian Conference on Image Analysis*, pp.61–70 (2009).
[65] Liew, C.F. and Yairi, T.: Generalized BRIEF: A Novel Fast Feature Extraction Method for Robust Hand Detection, *Proc. 22nd International Conference on Pattern Recognition*, pp.3014–3019 (2014).
[66] Bach, F.R., Lanckriet, G.R.G. and Jordan, M.I.: Multiple Kernel Learning, Conic Duality, and the SMO Algorithm, *Proc. 21st International Conference on Machine Learning*, pp.1–8 (2004).
[67] Gönen, M. and Alpaydın, E.: Multiple Kernel Learning Algorithms, *Journal of Machine Learning Research*, Vol.12, pp.2211–2268 (2011).

Chun Fui Liew received his Bachelor's degree from Nanyang Technological University, Singapore in 2009, M.S. degree from National University of Singapore in 2011, and M.E. degree from the University of Tokyo, Japan in 2013. He is currently pursuing Ph.D. degree in Aerospace Engineering at the University of Tokyo. His research interests include UAV robotics, machine learning, and computer vision. He is a student member of the IEEE.

Takehisa Yairi received his M.E. and Ph.D. degrees from the University of Tokyo, Japan in 1996 and 1999 respectively. He is currently a full-time Associate Professor with the Graduate School of Engineering in the University of Tokyo. His research interests include data mining, machine learning, mobile and space robotics. He is a member of The Japanese Society for Artificial Intelligence (JSAI) and The Robotics Society of Japan (RSJ).

(Communicated by *Tyng-Luh Liu*)