Multilayer Architectures for Facial Action Unit Recognition

Tingfan Wu, Nicholas J. Butko, Paul Ruvolo, Jacob Whitehill, Marian S. Bartlett, and Javier R. Movellan

***Abstract*—In expression recognition and many other computer vision applications, the recognition performance is greatly improved by adding a layer of nonlinear texture filters between the raw input pixels and the classifier. The function of this layer is typically known as feature extraction. Popular filter types for this layer are Gabor energy filters (GEFs) and local binary patterns (LBPs). Recent work [1] suggests that adding a second layer of nonlinear filters on top of the first layer may be beneficial. However, it is unclear what is the best architecture of layers and selection of filters. In this paper, we present a thorough empirical analysis of the performance of single-layer and dual-layer texture-based approaches for action unit recognition. For the single hidden layer case, GEFs perform consistently better than LBPs, which may be due to their robustness to jitter and illumination noise as well as to their ability to encode texture at multiple resolutions. For dual-layer case, we confirm that, while small, the benefit of adding this second layer is reliable and consistent across data sets. Interestingly for this second layer, LBPs appear to perform better than GEFs.**

***Index Terms*—Action unit recognition, facial expression recognition, Gabor energy filters (GEFs), local binary patterns (LBPs).**

I. Introduction

EXPRESSION recognition systems are organized as a serial pipeline. The first layer in this pipeline detects and normalizes (registers) face images. The input to this layer is an image and the output is a collection of patches with the detected faces normalized to a common size. The second layer applies a bank of nonlinear filters to the image patches found by the previous layer. The goal of the second layer is to make the mapping from images to expression categories easier to separate by a classifier and to filter out irrelevant information, e.g., jitter or illumination noise. The final layer consists of a statistical classifier that converts the output of the filter bank into numbers representing the presence or absence of target facial expressions, e.g., multinomial logistic regression or a support vector machine (SVM).

Manuscript received June 8, 2011; revised November 9, 2011 and February 26, 2012; accepted March 6, 2012. Date of publication May 11, 2012; date of current version July 13, 2012. Support for this work was provided by NSF Grants IIS-0905622, IIS-1002840-SoCS and NSF IIS-INT2-0808767. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. This paper was recommended by Associate Editor M. Pantic.

The authors are with Machine Perception Laboratory, University of California San Diego, La Jolla, CA 92093-0440 USA (e-mail: ting@mplab.ucsd.edu; nick@mplab.ucsd.edu; paul@mplab.ucsd.edu; jake@mplab.ucsd.edu; marni@mplab.ucsd.edu; movellan@mplab.ucsd.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2012.2195170

Systems differ in the type of filters used in the intermediate layer. Geometric approaches utilize image pixel values exclusively as a means to extract the relative location and shape of a set of facial landmarks (e.g., corners of the mouth, brows) [2]–[6]. Texture-based approaches [7]–[9] use filters that describe the local texture information in the face image patches, without explicitly representing landmark locations.

Historically, the computer vision community focused first on geometric approaches perhaps because they closely matched our intuitions of how humans perceive faces, and because it was thought they would be less sensitive to changes in illumination and pose than texture-based approaches. Evidence has accumulated over the past 10 to 20 years that texture-based approaches perform consistently better than geometric approaches [10]–[12]. The reason is that in many cases, encoding the location and shape of facial landmarks turns out to be more difficult than directly encoding the target facial expressions. Moreover, some expressions (e.g., AU6 and AU7 in the Facial Action Coding System (FACS)) are very similar in shape but different in appearance, e.g., furrow lateral to the eyes in AU6 but not in AU7 [12]. In practice, current expression recognition systems are predominantly texture based or use geometry information to marginally boost the performance of a texture-based system. Pure shape-based approaches have for the most part been abandoned.

The simplest texture-based approaches send the pixel values from the face images directly to the statistical classifiers. It is clear by now that such systems are consistently outperformed by architectures that have an intermediate layer of texture filters between the pixel values and the statistical classifier [13], [14]. Two of the most popular texture filters in facial expression recognition are Gabor energy filters (GEFs) [15], and local binary patterns (LBPs) [9]. GEFs became of interest to the computer vision community in part due to the fact that they are mathematical models of the complex cells in primary visual cortex in the brain [15]. Some of the most successful expression recognition systems to date use banks of GEFs as their primary representation [7], [16]–[18]. LBPs originated in the computer vision literature on texture analysis [9] and since then have become popular in the facial expression recognition literature [14]. Interestingly, the basic kernels used in LBP filters are reminiscent of high spatial frequency, high bandwidth Gabor filters (see Section II-D).

A. Shallow versus Deep Architectures

In the language of neural networks, current expression recognition architectures are predominantly "shallow" or have a

"single hidden layer": The input layer is the set of raw pixel intensities, the hidden layer is a bank of nonlinear texture filters, and the output layer is a single-layer classifier. Shallow architectures are currently dominant in the machine learning community due in part to the great popularity and success of two shallow machine learning algorithms: SVMs and boosting [19]–[21]. However, recent years have seen a renewed interest in architectures that use more than one hidden layer (i.e., deep networks). This interest was rekindled partially by the development of efficient learning algorithms for training multilayer (deep) networks [22]. In computer vision deep architectures proved successful in problems such as digit recognition [23] and are now seeing a comeback [24], [25]. Interestingly, the winning architecture for the 2011 Facial Expression Recognition and Analysis (FERA) challenge [26] utilized a two-hidden layer architecture, named Local Gabor Binary Pattern (LGBP). This architecture applies LBP texture filters to the output of a layer of GEFs.

In this paper, we present a thorough empirical analysis of the performance of single-layer and dual-layer texture-based approaches on a difficult facial expression recognition task: FACS Coding [27]. In particular, we analyze how these different approaches generalize within and between data sets and how they resist perturbations due to local changes in illumination and registration jitter. The paper is organized as follows: First, we describe the system we submitted to the FERA competition, which utilized a single hidden layer architecture and placed second. In the second part of the paper, we describe a postcompetition analysis of texture-based architectures. In particular, we focus on the comparison between LBP and Gabor filters and the comparison between single and dual hidden layer architectures.

II. Competition System

The goal of the FERA competition was to automatically FACS code a collection of videos of actors making facial expressions. The FACS is a taxonomy of facial expressions based on a combination of 57 elementary components. These elementary expressions, known as action units (AUs) and action descriptors (ADs), can be seen as the "phonemes" of facial expressions: words are temporal combinations of phonemes. Facial expressions are spatial combinations of AUs. We entered the FERA competition with a previously developed FACS recognition system, named the computer expression recognition toolbox (CERT) [7], [17], [18]. Our focus was on analyzing how to best adapt an existing system, such as CERT, to generalize on a new data set. Fig. 1 describes the pipeline of the CERT system. In this section, we examine the various building blocks: their properties, common failure modes, and our efforts to optimize them.

A. Data Sets

FERA—The focus of the FERA 2011 (AU) competition was the GEMEP-FERA [28] data set. This data set consists of recordings of ten actors displaying a range of expressions, while uttering a meaningless phrase, or the word "Aaah." There are seven subjects in the training data, and six subjects in the test set, three of which are not present in the training set. The training videos came with frame-by-frame binary labels (present or not) of 36 AUs and AD50[1] (talking). Intensity ratings, location of apex, and reliability information were not available. Among the labeled AUs, only the prediction of 12AUs are evaluated on the test set (Table II).

The authors of this paper never had access to the labels of the test set. We had access to the test images themselves, but we chose not to label them in any way to estimate performance or to train our classifiers.

FFD07—This is the original training set used to develop CERT. It is a combination of the following databases: Ekman-Hager [13], Cohn-Kanade [29], MMI [30], M3 [31], and two nonpublic data sets collected by the United States government which are similar in nature to M3.

B. Training Frame Selection

A priori it is not clear whether using every image in the data for training will result in optimal classifier performance. For example, a large sequence of nearly identical frames may result in the classifier giving excessive weight to a particular instance of a facial expression. It is also possible that training on the onset/offset points, when the intensity of the expression is very low, may be counterproductive.

A common strategy to deal with this issue is to select only the apex frame from each labeled AU event, i.e., the frame in which the expression intensity is maximal [32]. Practically, as the competition data set came with only frame-by-frame binary labels instead of onset-apex-offset event information, an AU event was first localized by finding consecutive positive-labeled frames in a video, then the center frame of the sequence was selected as an approximation to the apex frame. However, this per-AU strategy ignores the fact that other co-occurring AUs may change the appearance of the target AU. For example, the brow appearances of AU1+AU4 (inner brow raiser + outer brow raise) differs greatly from the brow appearance of AU1 by itself. Therefore, instead of looking at each AU independently, we searched for unique AU-combinations in the video. When there were multiple disjoint sequences of an AU combination, only the center frame of the first event was selected.

We tried three different frame selection schemes: no frame selection, i.e., use all frames (NoFS); unique combinations among the 12 training AUs + AD50 (FS12) (as was done in the baseline paper [26]), and unique combinations among all 36 labeled AUs and AD50 (FS36).

Fig. 2 shows the cross-validation performance using the three different frame selection schemes. No significant performance difference were found between the three approaches. However, frame selection reduced the size of the FERA data set from 5264 (NoFS) images into 627 (FS12) and 934 (FS36). Having less training frames significantly sped up the training process giving up opportunities to test differences ideas. After all, we chose the FS36 approach in the rest of the paper for its slightly better performance (though not significant) and much shorter training time comparing to NoFS.

[1] AD50.

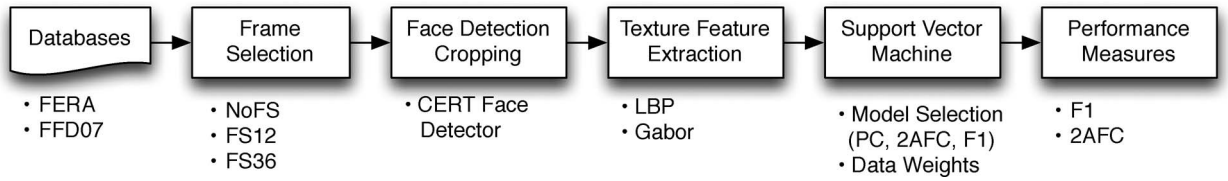

Fig. 1. Basic processing pipeline for approaches in this paper. Both the baseline method and CERT are special cases of the pipeline.

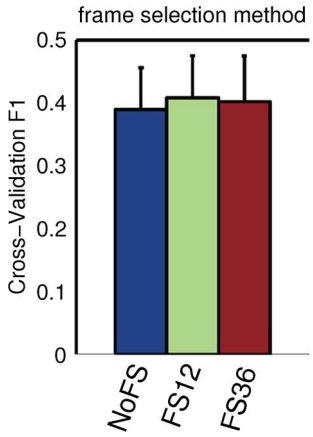

Fig. 2. Leave-one-subject-out cross-validation performance on the GEMEP-FERA training set using different frame selection methods. The F1 score is averaged across cross-validation folds and AUs.

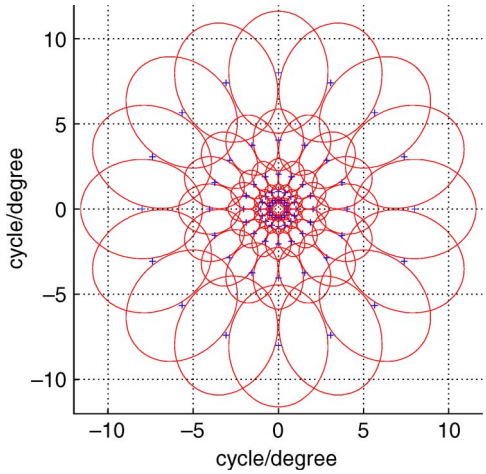

Fig. 3. Spectrum of the Gabor filter bank. The "+" centroids are the peak frequencies, and the ellipsoids are the half-magnitude contour of each filter. Pairs of a opposing centroids form one GEF, so 80 centroids yield 40 GEFs. We follow the convention that a face patch spans 4° of horizontal and vertical visual angle.

C. Face Detection and Registration

We used the CERT face detector and feature detectors [33] for face registration. Both detectors are of the Viola and Jones type [34] except that the face detectors discriminates face or non-face patches in an image while the feature detectors discriminate a face feature from the rest of the regions in a face. The CERT face detector detected all the faces in the competition training set with no false alarms except for the blank frames.

In the registration process, first the face detector finds a face in the image. Second, the feature detectors localize face features in the face, including eye corners, nose, and mouth corners. Third, the face is normalized using a planar least-square fitting procedure on the detected features (i.e., Procrustes analysis [35]). Finally, the aligned face is scaled into a 96×96 pixel matrix in which the intra-ocular distance is 48 pixels. The location of the detected facial features is discarded, and the 96×96 pixel image matrix is sent to the next layer of processing.

D. Texture Filters

The version of CERT we used for the competition employed a hidden layer representation with 40 GEFs (Fig. 3 plots the filter spectra), (five frequencies, eight orientations). This is slightly different from the standard version of CERT, which uses 72 filters [7], [17], [18]. In recent work, we found this new 40 filter version to work as well as the 72 filter version.

The baseline algorithm provided by the FERA organizers used an LBP image representation consisting of histograms of the 59 neighborhood-8 radius-1 uniform LBP values in nonoverlapping 20×20 pixel blocks of the 200×200 cropped face, so that there were 10×10 blocks. In our implementation, the LBP operated on smaller 96×96 cropped faces.

There are interesting similarities between Gabor filters and uniform LBP filters [36]. Both approaches are spatially local-oriented edge detectors with some robustness to translation and changes in illumination. Fig. 4 shows visualizations of both features, which highlights their similarity. LBPs achieve robustness to translation when their histograms are pooled over local blocks. GEFs achieve it by the fact that they are invariant to local phase. LBPs detect different types of local neighborhood (center surround, edge, or corner) by having different binary codes. GEFs characterize local neighborhoods by combining filters of multiple frequency and orientations. Both methods can identify edges of different spatial extent by adjusting their radius/scale. However, LBPs are typically implemented with single fixed radius. From a signal processing point of view, many of the LBP kernels can be seen as high frequency, high bandwidth bandpass filters.

Empirically, we found GEFs performed better than LBPs (see Section IV-A). Therefore, we used Gabor features in our competition submission.

E. Data-Weighted Support Vector Machines

SVMs [37] were used to map the output of the texture filters, either GEFs or LBPs, into AU categories. One of the main challenges we encountered was how to best combine FFD07, the prior data that had been used for training CERT, with the new FERA training set. The most straightforward way to adapt

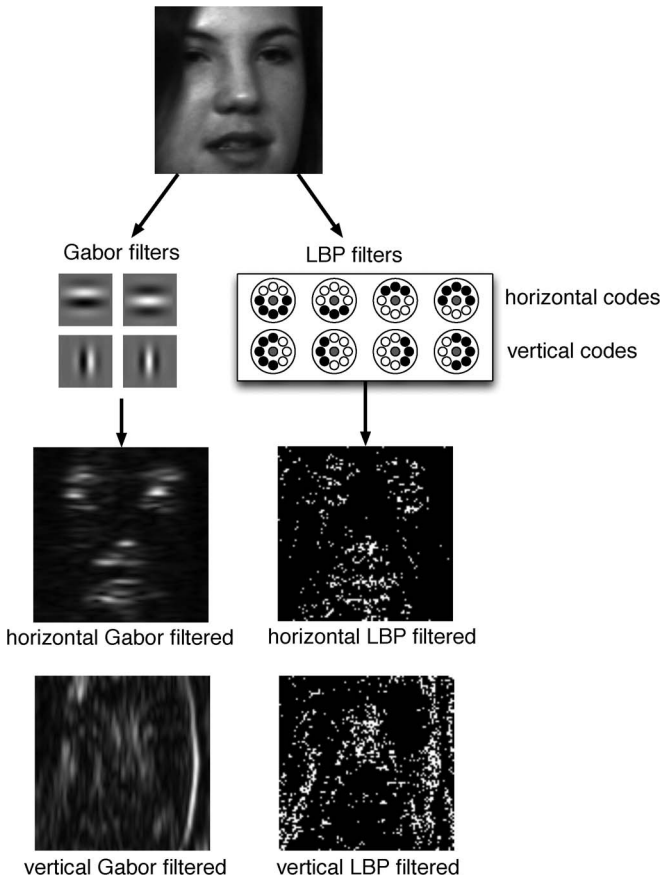

Fig. 4. Examples of Gabor and LBP filters and the filtered images. LBP encodes each pixel in the input by its neighborhood. Among the 59 possible uniform $\text{LBP}^{u2}_{8,1}$ codes, the figure shows a subset of codes that detect horizontal (top row) or vertical (bottom row) edges passing through the center pixel. In the LBP-filtered image, the white pixels represent the pixels encoded by the corresponding horizontal or vertical edge detecting LBP subsets.

CERT to the FERA data set is to combine FERA and FFD07 and retrain on the combined data set. However, this method has some potential disadvantages: The FFD07 data set may introduce information that is counterproductive for the FERA challenge. For example, FFD07 has a significant proportion of individuals of Asian and African descent, while the subjects in GEMEP-FERA are strictly European. This makes CERT more applicable to a diverse population but may deteriorate performance on the FERA data set, which is a nonrepresentative subset. Moreover, the FERA database is quite small (FS36: 934 images) after frame selection. Thus, the learning algorithm (SVM) could be easily overwhelmed by the instances from FFD07 (8000+ instances).

To address this problem, we developed a custom version of SVM with data weights that can be individually adjusted for each training example. Given training data $\{\mathbf{x}_i\}$, and labels $\{y_i\}$, the primal formulation of the data-weighted SVM is

$$\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_i \overbrace{c_i}^{\text{data-wt.}} \xi_i \tag{1}$$

$$s.t. \quad y_i(\mathbf{w}^T\mathbf{x}_i - b) \geq 1 - \xi_i$$
$$\xi_i \geq 0. \tag{2}$$

As in the traditional SVM formulation, $(\mathbf{w}, b)$ defines the hyperplane to be learned, ξ_i are the slack variables, and C is the master data fitness parameter; the added c_i are the data weights, controlling the fitness to each data instance. The data-weighted SVM reduces to standard SVM when $c_i = 1$ for all data instances. Both linear kernel and nonlinear radial basis function (RBF) kernel $(K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2})$ were used. For the linear kernel, the hyper-parameter is the regularization parameter (C). and for the nonlinear kernel, there is an additional parameter, inverse kernel width (γ).

For each AU, a binary SVM was trained with all the selected frames using the optimal hyperparameter selected for the AU. The two hyperparameters were selected from a candidate grid of values using cross-validation accuracy [38]. However, our targeted performance measure is F1 which has very different properties from other performance measures such as 2AFC (see Section II-F). We decided to try and optimize three different performance measures: percent correct (PC), two-alternative forced choice (2AFC), and F1 score.

F. Performance Measures

Most statistical classifiers produce real-valued scores that can be interpreted as evidence for the observed data belonging to one of two categories of interest, here referred to as the positive and negative categories. The sensitivity of the classifier depends exclusively on the statistical properties of the real valued score, not the threshold. However, many applications require making binary decisions. For example, the smile shutter feature in some digital cameras needs to decide whether to take a picture based on the evidence that a person is smiling. These decisions are typically made based on whether the real-valued scores provided by the classifiers pass a given threshold. In such cases the performance is a function of the sensitivity provided by the real valued scores, and of the threshold chosen for converting those scores into binary outputs. A system with good sensitivity may appear to perform poorly for a specific problem if the threshold is not chosen judiciously.

The area under the receiver operating curve (ROC) is a popular measure of performance used in the pattern recognition community [39]. An advantage of the ROC score is the invariance to the prior probabilities of the two categories, thus facilitating comparisons across data sets and categories with different priors. In addition, the ROC score is also invariant to monotonic transformations of the continuous-valued score provided by the classifiers. One problem with the ROC is that there is not a standard way to compute it. The actual value depends on the numerical integration scheme used to estimate the area under the ROC from a finite number of points. It is our experience that different methods can result in ROC scores that differ from each other by up to five percentage points. In psychophysics, a popular task for measuring the sensitivity of an observer independently of its bias is the 2AFC. In this task observers are presented with two randomly chosen examples of the two categories of interest. One of the examples is from the positive category and the other from the negative category. The goal of the observer is to choose the positive example. A well-known result from the theory of signal detectability

TABLE I
REPORTED PERFORMANCE ON THE FERA BLIND TEST SET. SEE SECTION III TEXT FOR DETAIL

Method	F1 Score			2AFC		
	ind	dep	all	ind	dep	all
Official random	.531	.471	.512	.500	.500	.500
Official LBP baseline	.453	.423	.451	.631	.611	.628
CERT unmodified	.569	.514	.550	.746	.692	.723
CERT retrained on FERA	.604	.539	.583	.759	**.753**	**.758**
Winner [36]	**.631**	**.610**	**.628**	**.763**	.751	.752

ind: test on new subjects not in the training set
dep: test on new videos of subjects seen in the training set
all: test on mixture of both "indep" and "dep" cases

shows that the expected PC in the 2AFC task equals the area under the ROC [39], [40]. For example, if an observer has an area under the ROC of 0.9 it has a 90% chance of correctly choosing the correct alternative in the 2AFC task. Hereafter, we will use the term "2AFC score" rather than "area under the ROC." The reasons are: 1) the 2AFC is very easy to compute (see Appendix); 2) it does not depend on numerical integration methods; 3) it has an intuitive interpretation as the probability of correctly categorizing a randomly chosen positive and negative example; 4) it can be interpreted as one of the many algorithms to estimate the area under the ROC.

The official evaluation metric for the FERA competition was the F1 score, a popular measure of performance in the document retrieval community. F1 is a function of the sensitivity of the system, the prior probabilities of the two categories, and the threshold used to make such decisions. A common misconception about F1 is that it favors low false alarm rates. This is not necessarily the case. If the system has low sensitivity (e.g., the random baseline in Table I), the F1 score is maximized by having a large false alarm rate, which may be undesirable in some applications. Finally, another popular measure of performance is the PC, e.g., the proportion of expressions correctly classified. This is also a function of the sensitivity of the system, the prior probabilities of the different categories, and the chosen threshold.

G. Hyperparameter Fitting Criteria

The target of the competition was to optimize the F1 score averaged across all the target AU categories in a generalization set. One question of interest to us was whether optimizing hyperparameters (e.g., kernel width, regularization constant) with respect to another performance measure (e.g., 2AFC, PC) yields a higher F1 score on generalization sets.

To this end, we compared the F1 generalization performance when the RBF-kernel SVM hyperparameters were optimized with respect to F1, 2AFC, and PC using double cross validation. Surprisingly, we found that optimizing hyperparameters with respect to 2AFC or PC resulted in better F1 score than optimizing with respect to F1 [see Fig. 5(b)]. Further investigation revealed why the F1 score was not a good parameter selection criterion. As Fig. 5(c) shows, the performance landscape for F1 is multimodal (two separate white regions) thus making hyperparameter selection difficult and suggesting greater expected variability in generalization tests. In addition to the typical "good parameter region" for SVMs [41] [see Fig. 5(d)–(f)], the

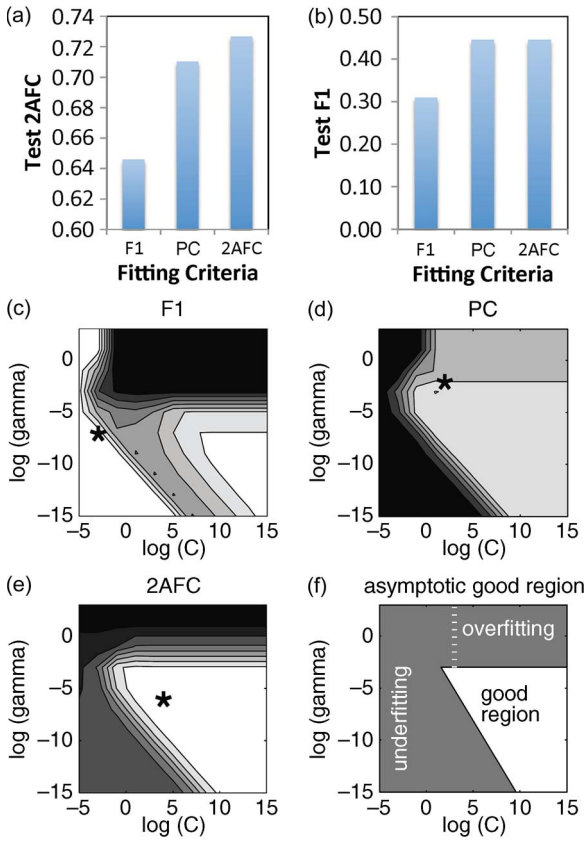

Fig. 5. (a) F1 and (b) 2AFC double cross-validation scores from SVMs with parameters optimized for different performance measure (c)–(e) An example of parameter selection with different performance measure. (f) asymptotic properties on nonlinear SVM model selection grid from [41]. The figures show the cross-validation performance surface for each grid search point of SVM parameters C and γ, the brighter pixels correspond to higher scores. The black "$\star$" denotes the final chosen parameter. Unlike PC and 2AFC, the surface of F1 scores is not unimodal, which makes parameter selection hard.

F1 score has additional peaks for small C parameters (large regularization). This leads to underfitting of the SVM weights and results in classification models that predict everything as positive. This may be due to the fact that for low sensitivity systems, F1 is optimized by using a threshold that encourages false alarms. The other two performance measures, 2AFC and PC, do not seem to suffer from this issue. Therefore, we used 2AFC to optimize hyperparameters in the rest of the paper. A previous study proposed that the F1 score could be preferable because algorithms were found that performed equally well with respect to the 2AFC but differed with respect to the F1 score [42]. Our results suggest that the F1 score may just be a noisier estimator of performance than 2AFC, potentially producing spurious differences between algorithms that are unrelated to their actual sensitivity.

III. COMPETITION RESULTS

Table I presents the official performance results of different algorithms, including, random, baseline, two of our methods, and the winning method. The random result is from a zero-sensitivity classifier which says "yes" for all frames. The official baseline results were provided by the FERA challenge organizers. We obtained second overall performance with

TABLE II
CROSS DATABASE COMPARISON OF 2AFC "OVERALL" SCORES

AU	CERT on M3 CV	CERT on FERA	CERT retrained on FERA	Baseline on FERA
1	.823	.747	.805	.790
2	.812	.745	.866	.768
4	.756	.719	.776	.526
6	.955	.835	.862	.657
7	.773	.701	.707	.555
10	.731	.681	.718	.597
12	.901	.815	.869	.724
15	.831	.643	.585	.563
17	.840	.639	.761	.646
18	.780	.649	.779	.610
25	.768	.760	.720	.593
26	.801	.691	.650	.500
Avg	.814	.719	.758	.628

respect to the F1 score, the target of the competition. However, we obtained the best overall performance with respect to the 2AFC score, which was not the target of the competition. Since the F1 score is sensitive to threshold while 2AFC is not, this suggests that the winning team chose thresholds better than we did.

A. Unmodified CERT Outputs

First, we applied CERT (previously trained on M3 before competition) directly on the FERA data set without any training or adjustment. In our original conference submission, the CERT generalization performance in all categories outperformed all of our approaches trained only on the FERA database, without having seen a single FERA image [44]. Table II displays the per-AU 2AFC scores, which includes previously published results on the M3 data set and the scores obtained on the entire FERA data set. The M3 results were obtained using single cross-validation methods (labeled M3CV), and thus they represent generalization within a data set, while the FERA results represent expected generalization to a new data set. When applied to the FERA data set, CERT took an average performance hit of only 9.1 percentage points. This is quite remarkable considering the different nature of the FERA data set when compared to the M3 data set. Fig. 6 shows a scatter plot of the AU by AU performance of CERT on the M3 versus the FERA data sets. The performance on the two data sets is highly correlated, which suggest the relative difficulties between AUs on the two database were comparable. However, there were some outliers: AU15, AU17, and arguably AU18. Performance on these AUs was worse than expected. One possible explanation is that for these AUs, the criteria utilized by the human coders of FERA may have been particularly different from the criteria used by the human coders of M3.

B. Retraining CERT on FERA

Table III describes the SVM data-weighting schemes for retraining CERT using FERA data: 1) Retrain on FERA only. 2) Retrain giving equal weight to FERA and FFD07. 3) Retrain giving 10 times more weight to FERA than FFD07.

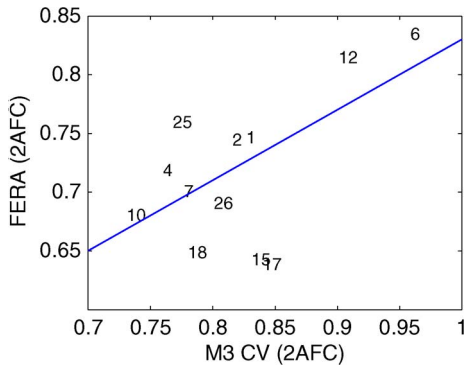

Fig. 6. CERT performance on the M3 versus the FERA data set for different AUs. The trend is linear except for AU 15, 17, and possibly 18.

TABLE III
THE WEIGHTING SCHEME USED IN THE COMPETITION

weighting scheme	c_i for FERA	c_i for FFD07
FFD07	0	1
FERA+FFD07*	1	1
10*FERA+FFD07	10	1
FERA	1	0

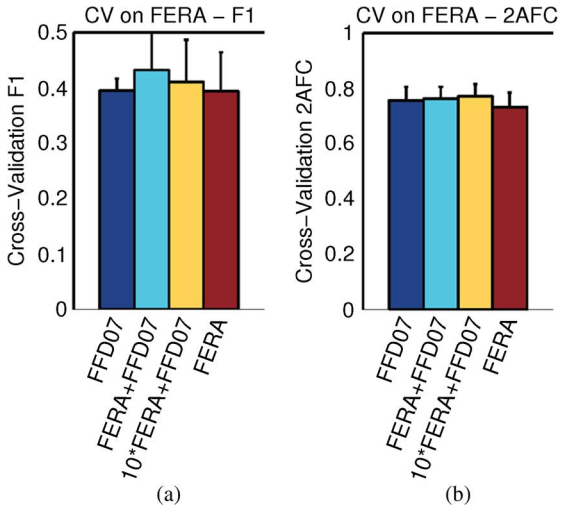

Fig. 7. Cross-validation performance of SVMs trained with different data weight settings.

Double cross validation [See Fig. 7(a) and (b)] indicated that best performance was obtained by retraining CERT on FERA and FFD07 with equal weights and thus our final competition system was based on that approach. Both the overall (Table I) and per-AU performance for our best competition system are shown in Tables II and IV). For subject-independent tasks, the performance gains of adding FERA were found in AUs that were particularly abundant, such as AU1, AU2, and AU4. For the subject-dependent case, the gain was particularly noticeable on the poor performing AUs. However, the overall performance improvements were marginally better. Similar performance results could have probably been obtained by retraining CERT on FERA alone.

TABLE IV
OFFICIAL TEST F1 SCORE OF CERT+FERA METHOD

	CERT			CERT+FERA		
AU	indep	dep	all	indep	dep	all
1	.496	.521	.503	.765	.399	.634
2	.689	.394	.564	.736	.485	.636
4	.596	.593	.595	.608	.590	.602
6	.804	.704	.777	.788	.683	.759
7	.579	.632	.601	.563	.660	.604
10	.502	.574	.528	.545	.598	.565
12	.832	.691	.781	.857	.789	.832
15	.188	.129	.161	.160	.246	.193
17	.542	.256	.456	.570	.328	.499
18	.203	.229	.214	.353	.334	.345
25	.836	.856	.844	.809	.821	.815
26	.565	.587	.575	.499	.533	.515
Avg	.569	.514	.550	.604	.539	.583

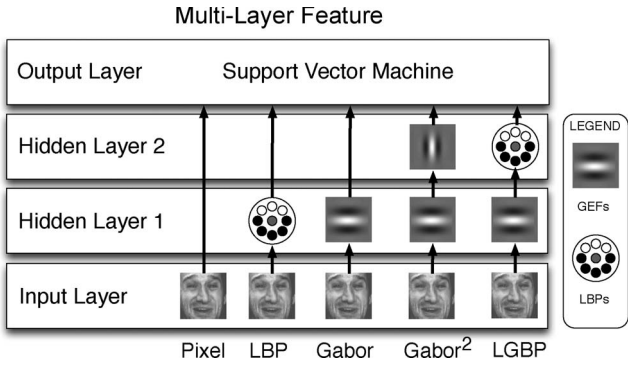

Fig. 8. List of features from single layer to three layers.

IV. POSTCOMPETITION ANALYSIS

There were some interesting differences between our system and the winning system. We used exclusively texture-based filters and the winning system used a combination of texture-based and geometric approaches. We were particularly intrigued however by the fact that the winning system used an architecture, named LGBP, consisting of two hidden layers rather than the single hidden layer architecture of our system. Post competition, we focused on comparing the performance of single-layer versus two-layer architectures. Here, we present the results of this analysis.

Fig. 8 displays the different architectures we investigated: The output layer of all the systems was a SVM. The simplest architecture used no hidden layers, i.e., the classifier was applied directly to the image pixels. The single hidden layer architectures used either LBPs or GEFs. The dual-layer architectures used either LBP on top of GEFs (this is commonly known as LGBP) or GEFs on top of a first layer of GEFs. We refer to this architecture as Gabor^2(G^2).

As these analyses were done post competition, the performance evaluation was based on leave-one-subject out double cross-validation methods on the FERA training set rather than the testing set. Therefore, the numbers are not directly comparable to those in the previous section. The performance metric was 2AFC averaged over all classified AUs. To speed up the training process, only those frames selected by FS36 were used for training.

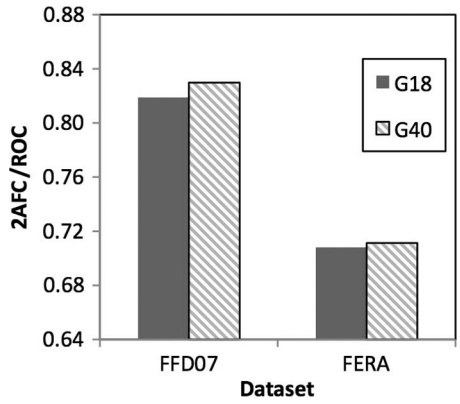

Fig. 9. Optimizing Gabor filter bank configuration: G40 (5 freq. × 8 orientations) or G18 (3 freq. × 6 orientations). (a) FERA dataset.

In addition to FERA data set, we also evaluated the different architectures on the FFD07 database using three-fold subject independent cross validation.

A. Optimizing Gabor, LBP, and LGBP Features

To perform a fair comparison of the different architectures we first optimized their parameters. For GEFs, we evaluated the following settings:[2]

G40—5 frequency (0.5, 1, 2, 4, 8 cycles/degree) and 8 orientations ($k\pi/8$, $k = 0, 1, \ldots, 7$). Fig. 3 plots the spectrum of these filters.

G18—3 frequency (2, 4, 8 cycles/degree) and 6 orientations $(0, \pi/6, \ldots, (5/6)\pi)$.

Fig. 9 shows the performance of these two settings in both data sets. Consistent with our prior experience, the results suggested that G40 was better than G18.

Regarding LBP/LGBP, an important parameter is the size and number of image regions over which histograms are computed. The prior literature seems to be somewhat divided on this issue. In [43], the authors partitioned the face into 16 regions organized as a 4 × 4 grid. In [1] the authors used a 7 × 6 grid and [45] used an 8 × 8 grid. In our analysis, we tried 4 × 4, 6 × 6, 8 × 8 and 10 × 10 grids. Fig. 10 shows the performance of LBP and LGBP on the two different data sets as a function of the grid size. In general, it appears that LBP benefited from using finer grid sizes. LGBP seemed to perform approximately the same regardless of the grid size. For further experiments, we used the 10 × 10 option. Next, we optimized the radius for LBP. The result is shown in Fig. 11. We found that for both LBP and LGBP, the performance deteriorated as radius increased though the gap between the smaller three radii were minimal. For further experiments, we used radius 1 pixel.

LGBP uses a first hidden layer of Gabor filters. We found that overall LGBP performed slightly better with the G40 filter bank than with the G18 filter bank. Therefore, for the rest of the experiments, LGBP was used with G40 filters in the first layer and 10 × 10 blocks in the second layer.

[2]We are using the standard that a face patch spans 4 × 4° of visual angle.

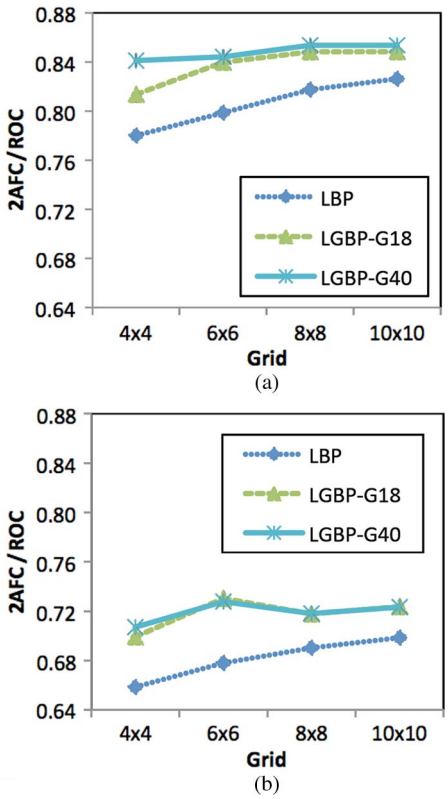

Fig. 10. Optimizing grid configuration and Gabor filters (G40 or G18) in LGBP and LBP. (a) FFD07 data set; (b) FERA data set.

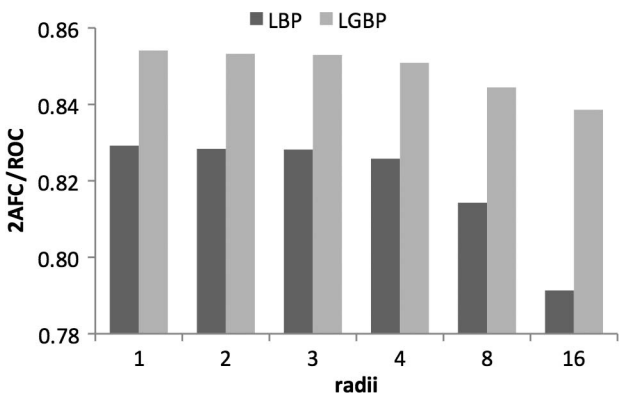

Fig. 11. Performance of LBP and LBGP for varying radii (pixels).

B. G^2 Architecture

Viola [46] pioneered the use of two layers of GEFs for representing image textures. The idea was that such representation could encode image textures that go beyond edges and bars. For example, it could encode the fact that a horizontal structure is made of tiny vertical bars. Fig. 12 shows an example of such structure. The contrast between the sclera and the iris creates vertical bars that excite vertically oriented Gabor filters. The edges between the teeth also excite the vertically oriented Gabors. A horizontally tuned Gabor applied on top of the vertically tuned Gabor then captures the fact that the teeth align horizontally. A similar phenomenon occurs with the eyes. The final result is a representation in which the teeth and eyes clearly stand out. Note as GEFs are nonlinear, the two layers of filters cannot be combined by using a single convolution operation.

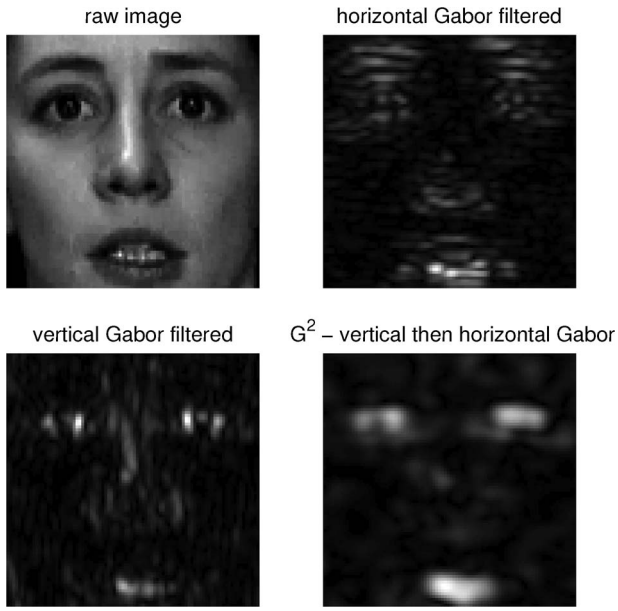

Fig. 12. Examples of Gabor energy and G^2 filtered face.

The G^2 architecture we tested used the G18 filter bank on top of the G40 filter bank. No further attempts were made to optimize this architecture.

C. Results

Fig. 13 shows the average performance of the different architectures on the FERA and FFD07 data sets. Results are presented in terms of 2AFC score using a cross-validation procedure within each data set. In all cases, the classifier was an SVM. We selected the best known kernel for each of the features. For LBP and LGBP, the histogram intersection kernel was used as in [43]. For Gabor and G^2, since linear and RBF kernel performed comparable in our experience and literature [8], so we chose the linear kernel for simplicity. For raw pixel, we used linear kernel for comparison.

Fig. 13 shows that the architectures with two hidden layers (LGBP and G^2) performed best, followed by one hidden layer architectures (Gabor and LBP), and lastly the architecture with no hidden layer. This trend is consistent in both data sets.

D. Resistance to Illumination Noise and Jitter

The original LGBP paper [47] suggested that LGBP performs better than Gabor and LBP due to the fact that it is less sensitive to local changes in illumination. GEFs are also supposed to be advantageous because of their relative insensitivity to local illumination and spatial jitter. To evaluate this hypothesis, we tested how the different architectures degrade as a function of jitter noise and illumination noise. The noise was added into both training and testing sets. Our expectation was that the architectures with two hidden layers would be the most resistant to noise, followed by one hidden layer and no hidden layer architectures.

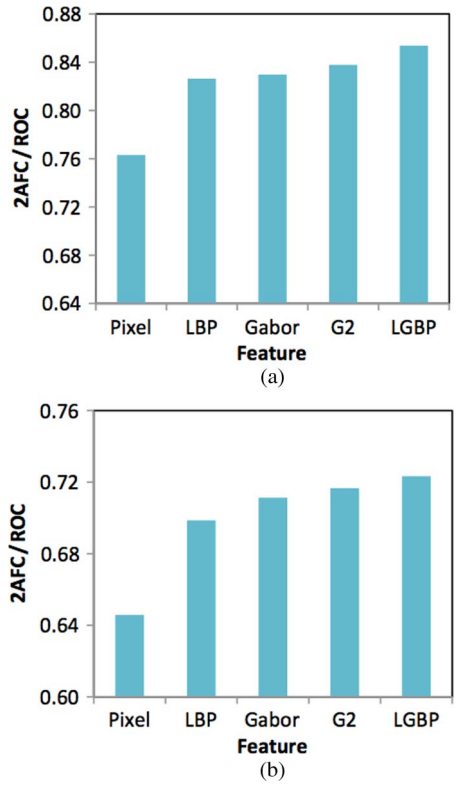

Fig. 13. Cross-validation performance of features on FFD07 and FERA data sets. (a) FFD07 data set; (b) FERA data set.

Fig. 14. Examples of simulated facebox jitter.

To simulate jitter, we added noise to the registered face box location. The noise was sampled randomly from a 2-D uniform distribution with support $[-9, 9] \times [-9, 9]$ pixels in the horizontal and vertical directions (see Fig. 14).

We also tested resistance to local illumination noise by adding Gaussian "spotlights" at random positions on the face (see Fig. 15). The spotlights were generated from the weighted average between the original image and a plain-white image using a 2-D isotropic Gaussian envelope with random centers and standard deviation $\mathbf{diag}([10, 10])$. In addition to simulated illumination variation, one could also use data sets with real varied illumination, such as the MultiPIE data set [48]). We leave this as future work.

Fig. 16 presents the cross-validation results on FFD07 database. As expected, the simplest architecture with no hidden layers was more vulnerable to jitter and illumination noise than those with hidden layers. Among the single hidden layer architectures, Gabor decayed slower than LBP in both types of noise. This suggested that Gabor features are a good choice for

Fig. 15. Examples of illumination noise of varying strength.

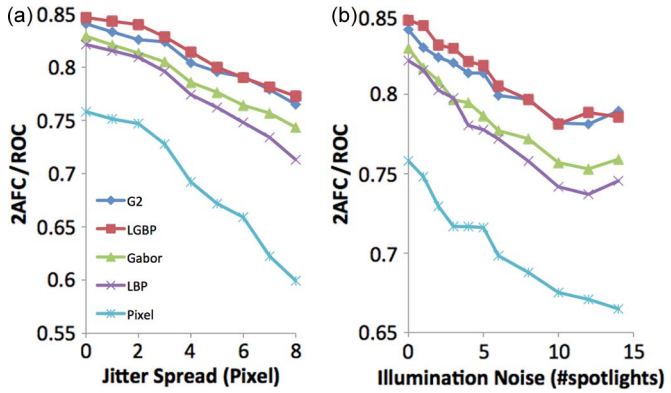

Fig. 16. Performance of features with (a) varying spreads of face registration jitter, and (b) varying intensities of illumination change.

first hidden layer. Best overall performance was obtained by the two hidden layer architectures (LGBP and G^2). The extra layer did improve the performance substantially. The second layer seemed to marginally improve the resistance of LBP to jitter and illumination noise. This can be seen by the fact that the LBP and LGBP curves tend to diverge as noise increases. However, the second layer did not seem to improve resistance to noise over the single-layer Gabor architecture. This can be seen by the fact that the curves of single-layer Gabor and of G^2 are parallel.

E. Multiresolution LBP

We wondered why Gabor filters performed consistently better in single-layer architectures than LBPs, particularly considering that Gabor and LBP are both basically oriented edge detectors. One difference is that Gabor filter banks typically utilize a range of peak spatial frequencies (Fig. 3), thus encoding multiple scales and resolutions. LBP on the other hand are typically applied with a single radius. Similarly, the advantage of LGBP over LBP could be partially attributed to the multiscale power provided by the first Gabor layer.

To explore these issues, we tested multiresolution LBPs with radii 1, $1 + 2$, and $1 + 2 + 3$ pixels. Our implementation of multiresolution LBP simply concatenated all the LBP feature vectors not unlike what we do with the Gabor architecture. Fig. 17 shows the results of the study. As expected, the performance of multiresolution LBP increased as the LBP feature pool increased. Surprisingly, the last set, $1 + 2 + 3$, even slighted surpassed the single-layer Gabor system, although the score was still lower than the two-layer architectures, G^2 and LBGP. However, as Fig. 17 shows, adding multiple resolutions to the second layer of LBPs did not seem to help. This may due to the fact that the Gabor layer in LGBP is already providing

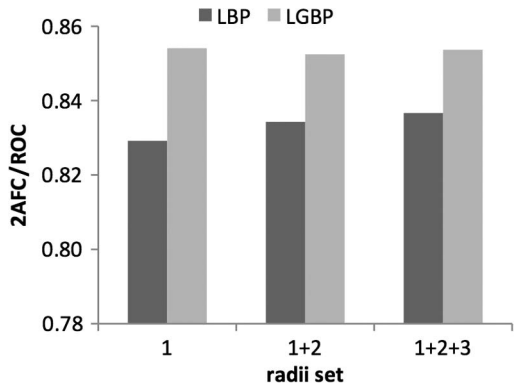

Fig. 17. Performance of LBP and LGBP for varying radii sets.

TABLE V
EVALUATION OF FEATURE ACROSS FERA AND FFD07 DATABASES. THE STANDARD ERROR SUGGESTS THE VARIABILITY OF THE RESULT ACROSS DIFFERENT AUS (a) LGBP; (b) G^2; (c) GABOR; (d) LBP; (e) PIXEL

Feature	Configuration	#Features
LGBP	59 bins * 100 blocks * 40	236000
Gabor	96*96 pixels * 40	368640
G^2	96*96 pixels * 18 * 40	6635520

multiresolution capabilities and that therefore adding these to the second layer is redundant.

F. Generalization Across Data sets

Table V shows the performance of the four different architectures when tested for generalization within and across data sets. When trained on FFD07 and tested on FERA, the performance of LGBP, G^2 and single-layer Gabors was similar, but LBP fell a bit behind. This suggests Gabor filters provide the best representation for the first hidden layer.

When trained on FERA and tested on FFD07, LGBP was the clear winner, followed by Gabor, and then G^2. We suspect the relatively poor performance of G^2 when generalizing from a small to a larger data set may be due to the fact that the version of G^2 we investigated has a very large dimensionality (6 635 520) in comparison to single-layer Gabors (368 640) and LGBP (236 000). Table VI describes how the number of features are calculated.

V. CONCLUSION

We compared single-layer and dual-layer architectures for texture-based expression recognition systems. We showed that dual-layer architectures provide small but consistent improvements in performance over single-layer systems. We show that for the first layer GEFs outperformed LBPs. Experiments suggested that this may be due to the fact that Gabor filters are more resistant than LBPs to noise caused by jitter or changes in illumination. However, for the second layer, LBP filters appear to perform marginally better than Gabor filters. We suspect that this may be due to the fact that the Gabor filter approaches we tried had many more parameters than the LBP approaches. Careful selection of the Gabor features in the second layer may make a difference. More work is needed to gain a better understanding of why two-layer architectures perform better and why Gabor filters are preferable for the first layer while LBP filters are preferable for the second layer.

TABLE VI
COMPARING THE NUMBER OF FEATURE OF DIFFERENT FEATURE TYPES

(a)

		test on FERA	test on FFD07
train on	FERA	.723±.032	.698±.028
	FFD07	.718±.030	.854±.017

(b)

		test on FERA	test on FFD07
train on	FERA	.717±.028	.650±.027
	FFD07	.721±.020	.838±.020

(c)

		test on FERA	test on FFD07
train on	FERA	.711±.029	.668±.021
	FFD07	.723±.021	.830±.022

(d)

		test on FERA	test on FFD07
train on	FERA	.699±.029	.629±.041
	FFD07	.685±.028	.828±.023

(e)

		test on FERA	test on FFD07
train on	FERA	.646±.024	.633±.029
	FFD07	.681±.027	.763±.028

APPENDIX

Pseudocode for computing 2AFC score.

```
% x is a vector of real valued numbers
% y is a vector of 0s and 1s
function s = Calc2AFC(x, y)
    x0 = x(y == 0);
    x1 = x(y == 1);
    n0 = length(x0);
    n1 = length(x1);
    s = 0;
    for k = 1: n0
        n = sum(x1 > x0(k)) + 0.5 * sum(x1 == x0(k));
        s = s + n/(n1 * n0);
    end
```

ACKNOWLEDGMENT

We thank the editor and anonymous reviewers.

REFERENCES

[1] X. Sun, H. Xu, C. Zhao, and J. Yang, "Facial expression recognition based on histogram sequence of local Gabor binary patterns," in *Proc. IEEE Conf. Cybern. Intell. Syst.*, 2008, pp. 158–163.

[2] T. Marks, J. R. Hershey, and J. R. Movellan, "Tracking motion, deformation, and texture using conditionally Gaussian processes," *Pattern Anal. Mach. Intell.*, vol. 32, no. 2, pp. 348–363, Feb. 2010.

[3] A. Ryan, J. Cohn, S. Lucey, J. Saragih, P. Lucey, F. De la Torre, and A. Rossi, "Automated facial expression recognition system," in *Proc. 43rd IEEE Annu. Int. Carnahan Conf. Secur. Technol.*, 2009, pp. 172–177.

[4] A. Ashraf, S. Lucey, J. Cohn, T. Chen, Z. Ambadar, K. Prkachin, and P. Solomon, "The painful face-pain expression recognition using active appearance models," *Image Vis. Comput.*, vol. 27, no. 12, pp. 1788–1796, Nov. 2009.
[5] P. Lucey, S. Lucey, and J. Cohn, "Registration invariant representations for expression detection," in *Proc. IEEE Int. Conf. Digit. Image Comput., Tech. Appl.*, 2010, pp. 255–261.
[6] M. F. Valstar and M. Pantic, "Fully automatic recognition of the temporal phases of facial actions," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 1, pp. 28–43, Feb. 2012.
[7] G. Littlewort, J. Whitehill, T. Wu, I. Fasel, M. Frank, J. Movellan, and M. Bartlett, "The computer expression recognition toolbox (CERT)," in *Proc. Autom. Face Gesture Recogn.*, 2008, pp. 1–2.
[8] C. Shan, S. Gong, and P. McOwan, "Facial expression recognition based on local binary patterns: A comprehensive study," *Image Vis. Comput.*, vol. 27, no. 6, pp. 803–816, May 2009.
[9] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 915–928, Jun. 2007.
[10] P. Michel and R. El Kaliouby, "Real time facial expression recognition in video using support vector machines," in *Proc. 5th ACM Int. Conf. Multimodal Interf.*, 2003, pp. 258–264.
[11] G. Littlewort, M. Bartlett, I. Fasel, J. Susskind, and J. Movellan, "Dynamics of facial expression extracted automatically from video," *Image Vis. Comput.*, vol. 24, no. 6, pp. 615–625, Jun. 2006.
[12] F. De la Torre and J. F. Cohn, "Facial expression analysis," in *Guide to Visual Analysis of Humans: Looking at People*. Berlin, Germany: Springer-Verlag, 2011.
[13] G. Donato, M. Bartlett, J. Hager, P. Ekman, and T. Sejnowski, "Classifying facial actions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 10, pp. 974–989, Oct. 1999.
[14] J. Whitehill, G. Littlewort, I. Fasel, M. Bartlett, and J. R. Movellan, "Toward practical smile detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 11, pp. 2106–2111, Nov. 2009.
[15] J. R. Movellan, "Optimal control of dynamic Bayes nets," UCSD, La Jolla, CA, 2005, vol. MPLab Tutorials.
[16] G. Littlewort, M. Bartlett, I. Fasel, J. Susskind, and J. Movellan, "An automatic system for measuring facial expression in video," *Image Vision Computing*, vol. 24, no. 6, pp. 615–625.
[17] M. Bartlett, G. Donato, J. R. Movellan, and J. Hager, "Face image analysis for expression measurement and detection of deceit," in *Proc. 6th Symp. Neural Comput.*, 1999, pp. 8–16.
[18] M. S. Bartlett, G. Littlewort, C. Lainscsek, I. Fasel, and J. Movellan, "Recognition of facial actions in spontaneous expressions," *J. Multimedia*, vol. 1, no. 6, pp. 22–35, 2006.
[19] V. Vapnik, *Statistical Learning Theory*. Hoboken, NJ: Wiley, 1998.
[20] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," in *Computational Learning Theory*. New York: Springer-Verlag, 1995, pp. 23–37.
[21] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *Ann. Stat.*, vol. 28, no. 2, pp. 337–407, 2000.
[22] G. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.
[23] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.
[24] J. Susskind, G. Hinton, J. Movellan, and A. Anderson, "Generating facial expressions with deep belief nets," *Affect. Comput., Emotion Model., Synth. Recogn.*, pp. 421–440, 2009.
[25] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "High-performance neural networks for visual object classification," IDSIA/USI-SUPSI, Manno, Switzerland, Tech. Rep. IDSIA-01-11, Feb. 2011.
[26] M. F. Valstar, B. Jiang, M. Méhu, M. Pantic, and K. Scherer, "The first facial expression recognition and analysis challenge," in *Proc. IEEE Int. Conf. Autom. Face Gesture Recogn.*, 2011, pp. 921–926.
[27] P. Ekman and W. Friesen, *Facial Action Coding System (FACS): A Technique for the Measurement of Facial Action*. Palo Alto, CA: Consulting, 1978.
[28] T. Banziger and K. R. Scherer, *Introducing the Geneva Multimodal Emotion Portrayal (GEMEP) Corpus*, K. R. Scherer, T. Banziger, and E. B. Roesch, Eds. London, U.K.: Oxford Univ. Press, 2010, ser. Blueprint for Affective Computing: A Sourcebook, Series in Affective Science, ch. 6.1, pp. 271–294.
[29] T. Kanade, J. Cohn, and Y. L. Tian, "Comprehensive database for facial expression analysis," in *Proc. IEEE Int. Conf. Autom. Face Gesture Recogn.*, 2000, p. 46.
[30] M. Pantic, M. Valstar, R. Rademaker, and L. Maat, "Web-based database for facial expression analysis," in *Proc. Int. Conf. Multimedia Expo.*, 2005, pp. 317–321.
[31] in prep M. Frank, M. Bartlett, and J. Movellan, The M3 Database of Spontaneous Emotion Expression (University of Buffalo), 2010.
[32] P. Lucey, J. Cohn, T. Kanade, J. SAragih, and Z. Ambadar, "The extended cohn-kanade dataset (CK+): A complete dataset for action unit and emotion-specified expression," in *Proc. CVPR Workshop Hum. Commun. Behav. Anal.*, 2010, pp. 94–101.
[33] M. Eckhardt, I. Fasel, and J. Movellan, "Towards practical facial feature detection," *Int. J. Pattern Recogn. Artif. Intell.*, vol. 23, no. 3, pp. 379–400, 2009.
[34] P. Viola and M. Jones, "Robust real-time face detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, May 2004.
[35] C. Goodall (1991). Procrustes methods in the statistical analysis of shape. *J. R. Stat. Soc.Ser. B, Methodological* [Online]. *53(2)*, pp. 285–339. Available: http://www.jstor.org/stable/2345744
[36] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.
[37] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Disc.*, vol. 2, no. 2, pp. 121–167, 1998.
[38] C.-C. Chang and C.-J. Lin, LIBSVM: A Library for Support Vector Machines, 2001. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm
[39] D. Green and J. Swets, *Signal Detection Theory and Psychophysics*. Los Altos, CA: Peninsula, 1966.
[40] S. Mason and N. Graham, "Areas beneath the relative operating characteristics (roc) and relative operating levels (rol) curves: Statistical significance and interpretation," *Q. J. Roy. Meteorol. Soc.*, vol. 128, no. 584, pp. 2145–2166, 2002.
[41] S. Keerthi and C. Lin, "Asymptotic behaviors of support vector machines with Gaussian kernel," *Neural Comput.*, vol. 15, no. 7, pp. 1667–1689, 2003.
[42] T. Simon, M. Nguyen, F. De La Torre, and J. Cohn, "Action unit detection with segment-based svms," in *Proc. IEEE CVPR*, 2010, pp. 2737–2744.
[43] T. Senechal, V. Rapp, H. Salam, R. Seguier, K. Bailly, and L. Prevost, "Combining aam coefficients with lgbp histograms in the multi-kernel svm framework to detect facial action units," in *Proc. IEEE FG*, 2011, pp. 860–865.
[44] T. Wu, N. Butko, P. Ruvolo, J. Whitehill, M. Bartlett, and J. Movellan, "Action unit recognition transfer across datasets," in *Proc. IEEE FG*, 2011, pp. 889–896.
[45] S. Moore, R. Bowden, and U. Guildford, "The effects of pose on facial expression recognition," in *Proc. BMVC*, London, U.K., 2009.
[46] J. De Boner and P. Viola, "Structure driven image database retrieval," in *Proc. Adv. Neural Inf. Process. Syst.*, 1998, pp. 866–872.
[47] W. Zhang, S. Shan, W. Gao, X. Chen, and H. Zhang, "Local Gabor binary pattern histogram sequence (LGBPHS): A novel non-statistical model for face representation and recognition," in *Proc. 10th IEEE ICCV*, 2005, pp. 786–791.
[48] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-pie," *Image Vis. Comput.*, vol. 28, no. 5, pp. 807–813, 2010.

Tingfan Wu received the B.Sc. degree in computer science from National Taiwan University, Taiwan, in 2004, and the M.S. degree from the University of California, San Diego (UCSD) in 2010, where he is currently pursuing the Ph.D. degree in developmental robotics and facial expression recognition.

Nicholas J. Butko received the B.Sc. degrees in computer science and cognitive science from the University of California, San Diego, in 2004, where he is currently working toward the Ph.D. degree.

His thesis research, conducted in the Machine Perception Laboratory since 2005, addresses the computational problems facing intelligent agents as they perceive, act, and interact in an uncertain and changing world. The emphasis here is to create theories that work in practice. To that end, he worked as an Intern at Intel in 2008 and 2009, to develop machine perception technologies for use in socially and physically aware perceptive tutoring systems.

Paul Ruvolo received the B.S. degree in computer science from Harvey Mudd College, Claremont, CA, in, 2003. He is a graduate student in the University of California, San Diego. His current research focuses on applied machine learning to analyze human behavior.

Jacob R. Whitehill received the B.S. degree in computer science from Stanford University, Stanford, CA, in 2001, and the M.S. degree in computer science from the University of the Western Cape, Bellville, South Africa, in 2006. He is pursuing the Ph.D. degree in computer science and engineering at the University of California, San Diego.

Marian S. Bartlett received the B.S. degree in mathematics from Middlebury College, Vermont, CA in 1988 and the Ph.D. degree in cognitive science and psychology from the University of California, San Diego (UCSD) in 1998.

She is a Research Professor at the Institute for Neural Computation, UCSD, where she codirects the Machine Perception Lab and leads the Computational Face Group. She studies machine learning, with application to face recognition and expression analysis. She has authored over 50 articles in scientific journals and peer-reviewed conference proceedings. Her book, Face Image Analysis by Unsupervised Learning, Kluwer, 2001, describes her work applying principles of information theory to face images. In collaboration with other members of the Machine Perception Lab, she has also developed a state-of-the-art system for automatic facial expression measurement that works in real time. Her thesis work was conducted with Terry Sejnowski at the Salk Institute on face recognition by independent component analysis. She has also conducted research on automatic recognition of facial expression with Paul Ekman and perceptual plasticity with V.S. Ramachandran. Her work spans machine vision and cognitive neuroscience, including papers in computer vision, visual psychophysics, neuropsychology, cognitive models of face perception, and the visuospatial properties of faces and American Sign Language. She also co-organized the 11th European Conference on Visual Perception, Bristol, England, and the 3rd International Conference on Development and Learning in San Diego, California, as well as numerous workshops at conferences such as Advances in Neural Information Processing Systems, and International Conference on Computer Vision.

Dr. Bartlett is Treasurer for the Neural Information Processing Systems Foundation, Associate Editor for Neurocomputing, and Director of Scientific Programs for the Temporal Dynamics of Learning Center at UCSD. In 2011, she was a Program Co-Chair for the IEEE International Conference on Automatic Face and Gesture Recognition.

Javier R. Movellan received the B.Sc. degree from the Universidad Autonoma de Madrid, Spain, in 1983, and the Ph.D. degree from the University of California, Berkeley (UC Berkeley), in 1990.

He founded the Machine Perception Laboratory (MPLab) at the University of California, San Diego (UCSD), San Diego, where he is currently a Research Professor. Prior to his UCSD position, he was a Fullbright Scholar at UC Berkeley, Berkeley and a Research Associate at Carnegie Mellon University, Pittsburgh, PA, from 1990 to 1993. The mission of the MPLab is to learn about intelligent behavior by developing systems that operate in the uncertain, but sensory rich conditions typically faced by the brain. His research interests include machine learning, machine perception, automatic analysis of human behavior, and social robots.