Pattern Recognition Letters 000 (2017) 1–9

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Facial expression recognition using temporal POEM features

Edwin Alberto Silva Cruz[a,*], Claudio Rosito Jung[b], Carlos Humberto Esparza Franco[c]

[a] *Electronics Engineering Department, Universitaria de Investigación y Desarrollo, Colombia*
[b] *Institute of Informatics, Federal University of Rio Grande do Sul, Brazil*
[c] *Unidades Tecnológicas de Santander, Colombia*

ARTICLE INFO

Article history:
Available online xxx

Keywords:
Facial expression recognition
LBP
POEM
Pattern recognition
Feature extraction

ABSTRACT

In this work, we propose a novel approach for Facial Expression Recognition (FER). We introduce the TPOEM (Temporal Patterns of Oriented Edge Magnitudes) features, which are volumetric extensions of the known POEM features by exploring adjacent frames and also temporal derivatives. To cope with the increase in the code length produced by TPOEM, we also present a novel coding scheme for the binary codes. TPOEM features are computed within non-overlapping image patches, and the final classification is achieved by combining the per-patch scores. The results showed increased accuracy compared to or better than state-of-the-art methods, while keeping the execution time low.

1. Introduction

Facial expression recognition is frequently performed on a daily basis by humans for interpersonal communication, and it is an important mechanism of recognizing emotions. However, automatic facial expression recognition (FER) is a relatively new field of research. One of the first relevant works was done by Kenji [16], applying Optical Flow algorithms. For the sake of comparison, the related problem of facial recognition has been studied since the seventies, with important works dated from back then [12,15]. Nonetheless, FER is a relevant field of research because the visual information contained in a facial expression can be used in applications such as evaluating retail efficiency [34] (e.g. evaluate facial expressions to determine the satisfaction of customers) and Human-Computer Interaction [3,42]. Despite the significant improvements in the past few years, there is still no consensus about the best approach in order to tackle the FER problem. A key issue is to devise discriminative features that allow fast and efficient FER, particularly when dealing with video sequences to capture temporal variations.

In this work, we present novel descriptors called TPOEMs (Temporal Patterns of Oriented Edge Magnitudes), which are built upon the known Patterns of Oriented Edge Magnitudes (POEMs) [44] by adding temporal information, which is important in the context of FER. Since the use of volumetric information in LBP-like coding increases the code length exponentially, we also propose a novel coding scheme for TPOEM, which is applied to non-overlapping patches of a face image. The final FER is obtained by combining the spatial patches using an adaptive weighted average, yielding results comparable to or better than state-of-the-art techniques in subject-independent validation tests with real time execution.

The remainder of this paper is organized as follows. In Section 2 we present a brief summary of existing facial expression recognition methods, while the proposed approach is presented in Section 3. Section 4 presents the experimental results, showing the accuracy of TPOEM using different configurations and comparisons with competitive approaches for different databases. Finally, in Section 5 we summarize the work and provide some perspectives and future work.

2. Related work

Facial Emotion Recognition (FER) from images is a typical problem of pattern classification: we want to label the input data (an image or video sequence) as one of the existing classes. There are two main types of features used in FER: the first one is based on geometric features, which are descriptors of facial geometry given a set of feature points that aim to characterize the shape of the face depending on the location of these points; the second type is based on appearance features, which describe textures within the image that may be descriptors of gestures, wrinkles and skin folds due to facial muscle movements. This division of features in these two categories is used by many authors, as shown by [6]. Next, we revise some FER techniques based on geometric or appearance features.

* Corresponding author.
E-mail address: esilvacruz@udi.edu.co (E.A. Silva Cruz).

http://dx.doi.org/10.1016/j.patrec.2017.08.008
0167-8655/© 2017 Elsevier B.V. All rights reserved.

2.1. Geometric-based features

Geometric-based features are used to describe the location and shape of facial features within the image. This is usually done by detecting and tracking fiducial points in important specific areas of the face, aiming to obtain the shape and size of the mouth, eyebrows and eyes (related to non-transient features), as well as wrinkles and furrows (related to transient features). The difference between transient and non-transient features is that the former do not appear all the time on the face, but they are temporarily present due to some muscular activation, while the latter are present all the time (although their shape, location and size differ due to the particular facial expression of the individual). To detect and track a set of fiducial points in the image/frame, most works have used AAMs (Active Appearance Models) [5] to describe the location and variation of the fiducial points. In order to describe the geometric characteristics of the face, distances between feature points, relative sizes, angles and shapes may be used to obtain a characteristic feature vector, which is used in the classification stage. Some works rely on the automatic detection and tracking of the fiducial points, but other systems need manual assistance or fully manual location of the fiducial points. Pantic and Rothkrantz [32] used a combination of fiducial point detectors and trackers based on the fact that each kind of detector performs well under particular circumstances, so that a combination of detectors provided a better overall tracking [36]. developed a facial expression recognition system in which the fiducial points were tracked using a specialized point tracker algorithm, and recognition was performed using a motion model. In [19], the user manually inputs the node of a Candide grid[1] in the first frame and then the system deforms the grid to match the images along the video sequence. In [4], an AAM was used with second-order minimization, while Ghimire and Lee [10] explored 52 facial landmarks that are detected and tracked using EGM (Elastic Graph Matching).

2.2. Appearance-based features

Appearance-based methods rely on the description of textures computed from the detected face. The general idea is that these textures are adequate descriptors to characterize gestures that represent facial expressions. The main advantage of using appearance-based methods is that they do not require the detection and tracking of fiducial points, whose calculation is usually costly and may need manual assistance. However, there is no consensus solution on which feature approach is more convenient, because works based on either approach have obtained similarly good results.

The main goal of appearance-based methods is to transform the pixel information of the face into a lower dimensionality representation that conveys information related to the different facial expressions. Some used techniques are PCA (Principal Component Analysis), ICA (Independent Component Analysis), kPCA (kernel-PCA), wavelets, Gabor features, sparse codes and LBP (Local Binary Patterns) [8]. implemented and tested several appearance-based representation methods, including PCA, ICA, LFA (Local Feature Analysis), LDA (Linear Discriminant Analysis) and Gabor wavelets, and the conclusion was that Gabor-based representations produced the best results. As such, Gabor wavelets have been used in several other facial expression recognition works, such as Zhang et al. [51] and Deng et al. [7]. However, the main drawback of Gabor wavelets is the high computational cost and memory requirements [26]. Lee et al. [21] presented an approach based on sparse representations, focusing on intra-class variation reduction and subject-independent scenarios.

[1] The Candide grid is a face mask consisting of 100 polygons (238 polygons for the Candide-2), used to the codification of faces by global and local Action Units.

A popular trend in appearance-based methods is the use of LBP and related codes, whose calculation is faster and the representation of facial features is adequate to perform the expression classification, as in [1,9,14,31,37]. In [53], VLBP was introduced. VLBP is an extension of regular LBP in order to include information about a number of neighboring frames, so that temporal transitions in the sequence are also considered. In [52], LBP-TOP, an orthogonal realization of LBP codes over the image sequence was implemented. Ramirez Rivera et al. [35] introduced the Local Directional Number (LDN) pattern, which also explores binary patterns along edge responses computed at different orientations.

More recently, methods based on deep learning architectures have become more popular. Such methods perform feature extraction implicitly (along the layers of the network), and work with raw image pixels. For instance, deep belief learning was explored by Liu et al. [23], whereas Convolutional Neural Networks (CNNs) were used in several works, such as [2,13,18,24,25,30].

Also, some methods combine geometric features with appearance features. A good recent example is the method by Happy and Routray [11], which encodes the local texture information at salient facial patches, extracted from fiducial points, using LBP patterns. Tariq et al. [41] explore overlapping image patches and extract local descriptors (e.g. SIFT) within the patches, encoding location information in super-vectors.

Although methods based on deep learning have been presenting the best accuracy (similarly to other areas of computer vision), they require a considerably large training set, and execution time might be an issue when using conventional CPU/GPU configurations. The goal in this work is to present an accurate yet fast method for FER, as presented next.

3. The proposed approach

The input is a video sequence containing a face, and each frame is split into non-overlapping spatial *patches*. Within each patch, a temporal neighborhood is used to compute the proposed descriptors, called Temporal Pattern Oriented of Edge Magnitudes (TPOEM), which are based on oriented spatial derivatives and also temporal derivatives. Each patch is then characterized by a set of histograms of TPOEM features, an and initial score is assigned per patch/orientation. These scores are then combined using an adaptive weighted average, such that patches with more discrimination power present larger weights. The overview of the proposed approach is shown in Fig. 1, and each step is presented in details next.

3.1. Temporal pattern oriented of edge magnitudes

POEM codification [43,44] was developed to combine desirable characteristics from micro-pattern methods and more globally oriented methods. More precisely, POEM was devised to be discriminative, robust, and computationally inexpensive in both terms of time and storage requirement.

The first step to extract POEM features is to calculate the gradient image at each pixel, which is discretized into a set of N equally spaced orientations θ_n, for $n \in \{0, \ldots, N-1\}$. Then, a local histogram of gradient orientations is computed over pixels within a spatial region, called cell, using the gradient magnitude as weight. The central pixel of the cell is then characterized by the accumulated magnitudes at each discretized orientation.

Finally, information about neighboring cells is encoded using LBP within another spatial region, called block. More precisely, such encoding at a given pixel p and orientation θ_n is given by

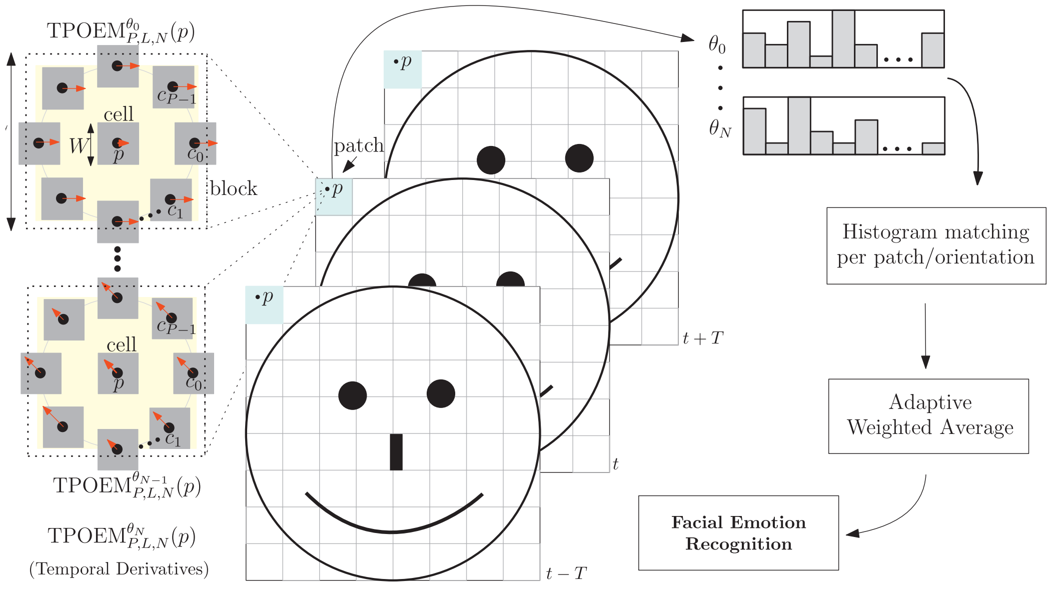
TPOEM$^{\theta_0}_{P,L,N}(p)$
cell
W
p
c_0
c_{P-1}
c_1
block
TPOEM$^{\theta_{N-1}}_{P,L,N}(p)$
TPOEM$^{\theta_N}_{P,L,N}(p)$
(Temporal Derivatives)
patch
$t+T$
t
$t-T$
θ_0
θ_N
Histogram matching per patch/orientation
Adaptive Weighted Average
Facial Emotion Recognition

Fig. 1. Illustration of the POEM-based FER classifier.

$$\mathrm{POEM}^{\theta_n}_{P,L,N}(p) = \sum_{j=0}^{P-1} f(s_{p,\theta_n} - s_{c_j,\theta_n})2^j, \quad (1)$$

where s_{p,θ_n} and s_{c_j,θ_n} are the cumulative gradient magnitudes along orientation θ_n for the pixel under analysis p and surrounding pixels c_j, which are the centers of their corresponding cells. Also, P is the number of neighboring pixels used in the LBP computation, L is the size of the block (that defines how spatially distant p is from c_j), and

$$f(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

is the unit step function. In [44], POEM codes are computed using the traditional 8 neighbors ($P = 8$), but here we let the number of neighbors to be generic. There is also an additional parameter W implicit to the POEM codification, which is the size of the cell used to obtain the cumulative magnitudes. The final POEM feature set at each pixel p is the concatenation of these unidirectional POEM values along all the discretized orientations

POEM codification [43,44] was successfully used in the context of Face Recognition (FR), which explores typically still images. Although Facial expression recognition (FER) can also be performed by the evaluation of a single frame or an image, it is intuitive to think that more information can be extracted from a video sequence, since a facial expression is the temporal transition from a neutral stance and a peak expression. As in the Volumetric LBP (VLBP) [52], the core idea is extend the neighborhood in which the LBP codes are computed by adding volumetric (spatio-temporal) information. However, instead of using a spherical neighborhood as proposed in [52], we present a neighborhood that is suitable for a more efficient implementation by re-using information from past frames. We also include temporal derivatives in the computation of the code, which adds extra information about the dynamics of the facial expression.

Formally, let us consider a given pixel p under analysis at frame t. Let s_{t,r,θ_n} denote the accumulated gradient magnitude at frame t, orientation θ_n, at a pixel $r \in \{c_0, \ldots, c_{P-1}, p\}$ within a block with size L (as in the POEM coding), around (and including) the central pixel $r = p$. Given other two frames[2] $t-T$ and $t+T$, where T denotes the temporal spacing, we first find the spatio-temporal neighbors of p. For the current frame t, there are P neighbors stored in N sets $V^{\theta_n}_{p,t}$ given by

$$V^{\theta_n}_{p,t} = \left\{ s_{t,r,\theta_n} | r \in \{c_0, \ldots, c_{P-1}\} \right\}, \quad (3)$$

for $n \in \{0, \ldots, N-1\}$. For both surrounding frames $t-T$ and $t+T$, there are $P+1$ neighbors stored in sets $V^{\theta_n}_{p,t\pm T}$, given by

$$V^{\theta_n}_{p,t\pm T} = \left\{ s_{t\pm T,r,\theta_n} | r \in \{c_0, \ldots, c_{P-1}, p\} \right\}. \quad (4)$$

Finally, the set $\mathcal{V}^{\theta_n}_{p,t}$ containing all spatiotemporal neighbors of p at a given orientation θ_n is given by

$$\mathcal{V}^{\theta_n}_{p,t} = V^{\theta_n}_{p,t} \cup V^{\theta_n}_{p,t-T} \cup V^{\theta_n}_{p,t+T}, \quad (5)$$

i.e. it is obtained by concatenating all neighbors. As it can be observed, there are $3P+2$ samples in $\mathcal{V}^{\theta_n}_{p,t}$, which is the binary code length for each orientation θ_n.

Once the volumetric texture is obtained, we define the Volumetric POEM (VPOEM) at the central pixel p in frame t as

$$\mathrm{VPOEM}^{\theta_n}_{T,P,L,N}(p,t) = \sum_{q=0}^{(3P+1)} f(v_{\theta_n,q} - s_{t,p,\theta_n})2^q, \quad (6)$$

where $v_{\theta_n,q}$ is the $(q+1)^{th}$ element of $\mathcal{V}^{\theta_n}_{p,t}$, and f the unit step function defined in Eq. (2).

As mentioned before, a cylindrical region was chosen instead of the original spherical region proposed in [52]. The main reason for that is to keep computational cost under control: when a spherical region is used, the whole dynamic texture must be re-computed as the frame under analysis t changes. With the proposed neighborhood, t depends on $t-T$ and $t+T$, and when we advance time in steps T, the next frame $t+T$ will depend on t and $t+2T$. Since the cylindrical cross-sections of our volume are constant, all the accumulated gradient magnitudes at frames t and $t+T$ can be completely re-used, leading to an incremental procedure presented in Algorithm 1. It should be run for each pixel p and orientation θ_n, and for the sake of clarity these variables were omitted in the algorithm.

[2] The number of temporal neighbors can be increased, but at the cost of increasing the binary code length.

Algorithm 1 Pseudo algorithm for volumetric texture extraction at a given pixel p and orientation θ_n.

```
1:  procedure VPOEM
2:      $t \leftarrow T$
3:      $V \leftarrow V^{\theta_n}_{p,t}$ (Eq. (3))
4:      $V_{\pm} \leftarrow V^{\theta_n}_{p,t\pm T}$ (Eq. (4))
5:      $\mathcal{V} \leftarrow \mathcal{V}^{\theta_n}_{p,t}$ (Eq. (5))
6:      Compute VPOEM using Eq. (6)
7:      repeat
8:          $t \leftarrow t + T$
9:          $\mathcal{V} \leftarrow \mathcal{V} - V_{-}$
10:         Compute $V_{+}$ using Eq. (4)
11:         $\mathcal{V} \leftarrow \mathcal{V} \cup V_{+}$
12:         Compute VPOEM using Eq. (6)
13:     until end of video sequence
```

Another way to include temporal information is in the computation of the POEM codification itself: instead of computing the oriented gradients just in a set of spatial orientations θ_n, $n \in \{0, \ldots, N-1\}$, we also incorporate a temporal derivative. For the sake of compactness in the notation, let θ_N denote such temporal dimension.

The spatial information encoded in the accumulated magnitudes is computed based on sums the cells. Although a analogous process could be employed for the time derivatives, using a larger temporal window would probably mix information from different facial expressions into the same code. Hence, we decided to use a forward difference scheme to encode temporal derivatives, so that the input to the LBP code in the temporal orientation θ_N is given by

$$s'_{t,p,\theta_N} = I_{t+T,p} - I_{t,p}, \tag{7}$$

where $I_{t,p}$ is the intensity of pixel p at frame t.

The final descriptor, Temporal Pattern Oriented of Edge Magnitudes (TPOEM) is defined as the concatenation of the VPOEM with the additional information provided by the temporal derivatives:

$$\text{TPOEM}^{\theta_n}_{T,P,L,N} = \begin{cases} \text{VPOEM}^{\theta_n}_{T,P,L,N} & \text{if } n < N \\ \sum_{j=0}^{P-1} f(s'_{t,c_j,\theta_N} - s'_{t,p,\theta_N})2^j & \text{if } n = N \end{cases}. \tag{8}$$

For computing TPOEM, the same parameters (N, W, L, P) used for POEM are needed, plus the temporal parameter T. However, the number of possible codes in TPOEM is much larger than POEM (2^{3P+2} versus 2^P using a direct coding scheme), which can be high even for relatively small values of P as noted in [53]. The use of uniform coding (codes for which there are at most two bitwise transitions from 0 to 1 or vice-versa when traversed circularly), as used in [53], drastically reduces the length of the final histogram: for B bits, there are only $B(B-1)$ uniform codes, and all non-uniform codes are assigned to the same bin. That means that computing TPOEM with $P = 8$ would lead to $\approx 67M$ raw codes compared to only 650 uniform codes. Our hypothesis is that such reduction also affects the discrimination of the volumetric texture, since different textural patterns may generate the same uniform code. To cope with this issue, we propose a novel coding scheme, described next.

3.2. A new coding scheme

Our coding scheme is based on two-stages: (i) extension of uniform coding to better discriminate volumetric textures, and (ii) code clustering to reduce the number of bins in the final histogram. As mentioned before, the traditional definition of uniformity presents a drastic reduction in the total number of codes: $B(B-1)$ versus 2^B for raw codification. Clearly, the reduction rate increases as the number of bits B increase, so that many different raw codes are mapped to the same uniform code (leading to a possible loss in the discrimination power). Due to this observation, the concept of uniformity was redefined as a code having at most N_T transitions from 0 to 1 or 1 to 0 when circularly traversed, where N_T is an even number. Larger values of N_T yield more uniform codes (and more discrimination), while the opposite happens with smaller values for N_T.

If increasing the number of uniform codes is good to get better discrimination among volumetric textures, it also generates longer histograms. Since our histogram matching is performed within small patches (i.e. not many samples are available), the histograms tend to be sparse. To reduce the number of remaining uniform codes, a clustering scheme is also applied. Our hypothesis was that for a wide variety of textures, several uniform codes (with the extended definition of uniformity) could represent the same texture, and hence being redundant. To test this hypothesis, we selected a set of random videos (378) from Youtube, with a total duration of approximately 1000 min. Small variations of the videos were obtained by the addition of noise and by cropping, to increase the database size, for a total of approximately 5M frames. The videos did not include more than one scene, so to guarantee smooth transitions between frames (and roughly uniform temporal texture). TPOEM codification was performed for closely located pixels along all the videos (called groups), and the co-occurrence of each pair of codes within the same group was stored in a similarity matrix M, normalized by the most frequent code between the two under analysis (to account for the uneven occurrence of the codes).

Large entries in M indicate raw codes that occur together for the same volumetric textural pattern. In this work, we first grouped together all pairs of codes (i,j) for which $M_{ij} > T_s$, where T_s is a similarity threshold. This pairwise strategy is then propagated hierarchically based on associativity, clustering together groups with elements in common: if (i,j) and (j,k) are paired, then (i,j,k) are clustered into a single bin in the final codification scheme. As for the threshold T_s, it is selected adaptively such the final histogram contains approximately a target number of clusters N_b is achieved in the end of the process.

3.3. Facial expression recognition using TPOEM

In this work, we tackled the 6-expression problem, related to the six basic emotions (Anger, Disgust, Fear, Happiness, Sadness, and Surprise), as well as the 7-expression problem, which refers to the six basic emotions plus Neutral stance. Given an input video sequence, the face is identified at each frame and resized to a standard resolution (128 × 128 pixels). The TPOEM feature computation given by Eq. (8) produces, for each pixel p, a set of $N + 1$ LBP-like codes $\text{TPOEM}^{\theta_n}_{P,L,N}(p)$, one for each orientation θ_n (including the temporal orientation). They encode local information, but the extent of this locality depends on the block size L and cell size W. To incorporate more spatial information, the whole image is divided into a set of non-overlapping patches disposed along a rectangular grid on the face region (64 in the total, using an 8×8 grid), as illustrated in Fig. 1. Within each of these patches, we compute the histograms at each orientation individually, so that a total of $K = 64(N + 1)$ oriented patches are generated for each image.

In [43,44], a similar approach was used, but the histograms for the patches were concatenated into a single (large) feature vector that characterize the whole face, and dimensionality reduction techniques based on whitening and PCA were used. We tested their approach for FER, but WPCA dimensionality reduction was able to reduce the dimensionality of data by 50–60% approximately, but the transformed dataset did not provide better classification rates, and in some cases the classification rates were drastically lowered. In this work, we performed an individual matching process for each patch (for which the feature vector presents lower dimensionality), and then combined the results into a global distance value for the whole facial expression.

Let us consider a training set formed by histograms $\mathrm{Tra}_{i,k,n}$, related to the n-th training sample, k-th patch and i-th expression. The prototype histogram $\mathrm{Av}_{i,k}$ for each patch k and i-th expression is given by the mean per-expression histogram, i.e.

$$\mathrm{Av}_{i,k} = \frac{1}{N_i}\sum_{n=1}^{N_i}\mathrm{Tra}_{i,k,n}, \tag{9}$$

where N_i is the number of training samples of the i-th expression.

In the test phase, we retrieve the histogram $\mathrm{Val}_{k,m}$ related to test sample m at patch k, and compare it to the class prototypes $\mathrm{Av}_{i,k}$ of the same patch and all expressions i. Although several histogram-based distances can be used, our experiments showed that the χ^2 distance, given by

$$D_{i,k,m} = \sum_x \frac{\left(\mathrm{Av}_{i,k}(x) - \mathrm{Val}_{k,m}(x)\right)^2}{\mathrm{Av}_{i,k}(x) + \mathrm{Val}_{k,m}(x)}, \tag{10}$$

is a good choice. In Eq. (10), x corresponds to each element within the TPOEM code (i.e. each histogram bin).

To find the overall distance $d_{i,m}$ between sample m and a class prototype i, we use a weighted average of the histogram distances per patch, i.e.

$$d_{i,m} = \sum_{k=1}^{K} w_k D_{i,k,m}, \tag{11}$$

recalling that $K = 64(N+1)$ is the total number of oriented patches, and w_k are the corresponding weights. It is important to note that the average of χ^2 histogram distances was also used by Zelnik-Manor and Irani [47] to compare and cluster temporal events within long continuous video sequences. However, each histogram comparison in their approach is related to temporal features computed at different scales, while our histograms relate to different spatial regions of the image.

Finally, the winning class i^*_m for test sample m is assigned based on the minimum distance, i.e.

$$i^*_m = \underset{i}{\operatorname{argmin}}\{d_{i,m}\}. \tag{12}$$

To define the weights w_k, we first note that some patches on the face provide more information than others [39]. We then randomly split the training data (CK+ dataset) into two disjoint sets Ω_1 and Ω_2, split in a subject independent manner (i.e. subjects in one set do not appear in the other). They have approximately the same size (might not be exactly equal due to the number of subjects and samples per subject), and Ω_1 is used to build the class prototypes $Av_{i,\,k}$, according to Eq. (9). For each training sample m of the second set Ω_2, for which the class label $c_m \in \{1,\ldots,I\}$ is known (here, $I = 6$ for the problem without the neutral expression, and $I = 7$ when the neutral expression is used), we compute the distance to the class prototypes at each patch according to Eq. (10).

For relevant regions, $D_{i,k,m}$ should produce low average values for the correct class (i.e. when $c_m = i$), and higher values for the remaining classes ($c_m \neq i$). Hence, the normalized error for the correct class label at each patch k, given by

$$\epsilon_k = \sum_{m\in\Omega_2} \frac{D_{c_m,k,m}}{\sum_{i=1}^{I} D_{i,k,m}}, \tag{13}$$

would produce the ideal value zero for relevant blocks, and larger values for non-discriminative blocks. To obtain the weight w_k of each patch, we choose a monotonically decreasing function (w.r.t.

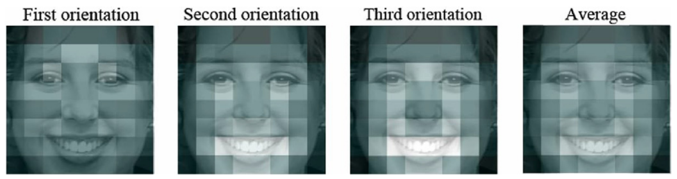

Fig. 2. Weights of the patches for three orientations, and their average (for the sake of illustration).

ϵ_k) and normalize the values, so that $\sum w_k = 1$. More precisely, the weights are given by

$$w_k = \frac{1}{\sum_{k=1}^{K} e^{-\alpha\epsilon_k}} e^{-\alpha\epsilon_k}, \tag{14}$$

where α controls the decay of the exponential function (set experimentally to 1). Note that this mapping is quite similar to the softmax function, with the inclusion of the decreasing monotonicity.

Assuming that the faces are mostly in a frontal pose, they can be considered symmetric w.r.t. the central vertical axis, so that the weights w_k are also symmetrized by averaging the original weights with a their horizontally flipped version. Fig. 2 shows a visual representation of the final weights in the three spatial orientations $\theta_n \in \{0, 2\pi/3, 4\pi/3\}$, where brighter patches carry more weight. As can be observed, regions around the mouth and the eyes are more important, as expected. On the other hand, regions on the boundary present mostly low weights, which is again expected since they tend to also comprise background pixels when the face is cropped.

4. Experimental results

This section shows the results obtained with the proposed approach, as well as comparisons with other state-of-the-art methods. It is worth noticing that comparisons with other FER methods is not an easy task, since each approach uses a different set of databases and/or validation strategies [48]. In this work, most of the results will be performed using the extended Cohn–Kanade (CK+) database [27], since it is the most popular among FER methods. Also, most FER techniques that use a subject-independent (SI) validation methodology, which is better for inferring the generalization of the classifier than subject-dependent (SD) ones, show their results for the CK+ database. Nevertheless, we also evaluate the MMI database [33] and a cross-database experiment using the KDEF (Karolinska Directed Emotional Faces) database [28]. In the experiments, we tackled the 6- and 7-expression classification problems and computed all results of the proposed method using a subject-independent 10-folded cross-validation. When overall results are reported, they are computed as the average of the per-class accuracies and not the average accuracy of all samples, which avoids biasing toward expressions with more samples in the databases. Although TPOEM explores more than one frame to classify a given sample, tests were performed using a moving temporal window, so that the accuracy was computed for each frame.

Regarding calculation costs, there are stages whose calculation costs are fixed, other stages whose calculation costs are almost constant, and yet another set of stages whose costs depend on the codification complexity. The fixed costs relate to frame grabbing, face detection and illumination correction. Mapping and classification have little dependency on the used parameters: mapping only requires μs time per code, and since the code length was adjusted to be “close to” 256 regardless the used parameters, the classification complexity does not change significantly. Since the coding scheme presented in Section 3.2 is based on hierarchical clustering, the number of clusters is reduced (typically by more than one) at each iteration. The target number of clusters is set as input in the procedure and used as a stopping criterion, and the actual number of cluster N_b returned by the procedure might be a little smaller.

The proposed method was implemented in Python, and tests were performed using a laptop computer with an Intel Core i7-4700 HQ, 2.4 GHz processor, 8GB RAM memory. The fixed and almost fixed costs take roughly 110ms in average according to our tests, so for $T = 5$ there are approximately 60ms left for codification in order to achieve real time calculation (recalling that only one out of five frames are processed). The average execution time per frame for our default method, TPOEM$_{5,5,16,1}$, was 26ms, allowing real-time processing for videos acquired at 30 frames per second.

4.1. Parameter setting

The computation of TPOEM requires a set of parameters, namely L, W, P and N. It should be noticed that P (neighborhood size) and N (number of orientations) impact the computation time and code length. Larger codes tend to produce better accuracy, but at a higher cost. Aiming to keep the complexity similar to other methods based on LBP, such as [38,53], we selected $P = 5$ neighbors a single ($N = 1$) gradient orientation (for the sake of comparison, $P = 8$ and $N = 3$ were used for POEM in [44]). The cell size W has no impact the cost (since the accumulated magnitudes are obtained using integral images), as well as the block size L.

To set these two parameters, as well as the temporal offset T, we perform a 10-folded cross validation procedure using the CK+ dataset, and run it ten times for each set of parameters in order to reduce the confidence interval. We evaluated the accuracy produced by changing each parameter separately, fixing the remaining ones. Although better results could be obtained by minimizing a target function that depends on all tested parameters, such approach would be computationally expensive (due to the chosen protocol), and could also overfit the parameters to training database (CK+). Some parameters are dependent on scale, namely L and T, which are scaled by resolution and FPS respectively. This was addressed by keeping a fixed size of 128×128 pixels for the detected face and proportionally scaling T if the FPS was different than 30.

As in [44], which was focused on FR, we concluded that varying the block size (except for very small or very large values) did not have significant impact on the accuracy, and used $L = 16$ as a default value (in fact, the average accuracies varied about 2% for $L \in [10,20]$). Regarding variable W, our experiments indicated that the accuracy rapidly decreases as W increases. This is expected because W defines a local region, so an increase in W reduces the characterization of local features which are important for the definition of the facial expressions. We selected $W = 3$ as our default value.

We varied the temporal offset T for estimating the derivatives, and results showed significant difference with $\alpha = 0.05$ for the interval $T < 4$ and $T > 10$ (by using $\alpha = 0.1$, the statistical difference interval increased to $T < 5$ and $T > 9$). Within the range $T \in \{5,6,7,8\}$, we chose $T = 5$ as our default parameter, since the yields the lowest lag (150ms for video sequences acquired at 30 FPS). Finally, uniform and rotational codification are included in the coding scheme, leading to a final TPOEM histogram with 177 bins. This set of parameters was fixed for all experiments shown in the remaining of the paper, unless explicitly mentioned otherwise.[3]

[3] To obtain this value, we evaluated the confusion matrix for the “modified 7-expression” problem (six basic emotions + contempt) from the original paper, and considered correct all misclassifications of any expression as contempt. Hence, it represents is a best-case scenario.

Table 1
Classification accuracy (in %) for our final method (TPOEM) and competitive approaches using the CK+ database.

Number of expressions	6-class	7-class	SD/SI	Partition
SPTS+CAPP [27]	83.15	–	N.A.	N.A.
Psycho-Visual [17]	95.3	–	N.A.	2 to 10 folds
Feature tracking [22]	87.43	–	SI	LSO
Sparse Representation FER [21]	90.47	–	SI	LOSO
Features of Salient Patches [11]	94.09	–	SD	10 folds
LDN$^G_{1.0,1.3,1.6}$ [35]	89.30	–	SI	10 folds
CNN-based [25]	91.46	–	SI	8 folds
CNN-based (C_{classE}) [24]	96.76	**95.79**	SI	8 folds
Collaborative [20]	96.12		SI	LOSO
POEM	95.02	90.27		
VPOEM$_{5,5,16,3}$	96.83	91.58	SI	10 folds
TPOEM$_{5,5,16,1}$	96.87	91.80	SI	10 folds
TPOEM$_{5,5,16,5}$	**97.39**	92.91	SI	10 folds

The bold values correspond to the highest classification rates for 6 and 7 classes including our methodology and the benchmark found in the state of the art.

4.2. Results and benchmarking

The default TPOEM scheme is given by TPOEM$_{5,5,16,1}$, obtained based on a compromise between accuracy and running time (it runs at 30FPS), as mentioned before. Table 1 contains the accuracy results for our default TPOEM method, as well as a comparison with other competitive approaches for the CK+ dataset [27], which is an extended version of the CK dataset with more samples and revised labels (labels in CK were not validated, so several sequences were not actual valid representation of the labeled expression).

The results were extracted from the respective papers, and in some of them, SI validation methodology was used (i.e. subjects in the test set were not used to train the model, as in our validation scheme), while in others it is SD (i.e. the same subject could be both in the training and test sets), or even unclear (marked as N.A. in the table). As noted by different authors [21,24,40,46], and also as explored later in this work, the same classifier tends to produce lower accuracy rates when SI validation is performed in comparison to SD. For instance, [46] reported an accuracy drop from 99.40% to 88.90% just by changing the validation strategy from SD to SI. In fact, SI validation provides a better measure of the generalization capabilities of a given classifier, and SD results might be viewed as upper bounds for the accuracy rates.

Compared to these competitive approaches, TPOEM$_{5,5,16,1}$ presents the best result (96.87%) among all methods for the 6-expression problem. It also presents the second best result (91.80%) for the 7-expression problem, being inferior only to the recent CNN-based method proposed by Lopes et al. [24], which includes a set of pre-processing steps (spatial/intensity normalization and synthetic samples for training). In particular, our method is considerably better than Lee et al. [21], which has been designed to deal with person-independent recognition as stated by the authors. It is also interesting to compare our results with Happy and Routray [11], which uses LBP features computed at the location of fiducial points. This approach is similar to ours: we compute TPOEM codes spatially distributed along the face, but within a fixed geometric grid instead of attaching them to fiducial points. We believe that the gain achieved by using a better appearance model (TPOEM instead of LBP) compensates the use of a coarser geometric model. Finally, Table 1 also shows that even better results can be obtained with TPOEM, but at the cost of higher computational cost. For instance, TPOEM$_{5,5,16,5}$ (using 5 orientations instead of only 1 in the default TPOEM) runs at 25 FPS, while TPOEM$_{5,5,16,1}$ runs at 37 FPS, enough to produce real time computation. This time includes frame grabbing, face detection, parameter extraction and classification, but the bulk of the work is done in the parameter extraction

Table 2
7-expression confusion matrix (%) using TPOEM$_{5,5,16,1}$ in CK+.7.

	Ang.	Dis.	Fea.	Hap.	Neu.	Sad.	Sur.
Ang.	**93.52**	0.00	0.00	0.00	6.48	0.00	0.00
Dis.	0.00	**91.58**	2.35	2.35	3.62	0.00	0.00
Fea.	0.00	0.00	**89.33**	0.00	7.33	0.00	3.33
Hap.	0.00	0.00	0.00	**100.00**	0.00	0.00	0.00
Neu.	2.21	1.07	1.43	0.00	**90.64**	3.93	0.71
Sad.	0.00	0.00	0.00	0.00	13.00	**87.00**	0.00
Sur.	0.00	0.00	0.83	0.00	4.42	0.00	**94.75**

The bold values are the diagonal of the confusion matrix (i.e. the correct classification per class).

Table 3
6-expression per-class classification accuracy (%) using our method (TPOEM$_{5,5,16,1}$) and competitive approaches for MMI database.

Method	Ang.	Dis.	Fea.	Hap.	Sad.	Sur.	Mean
[45]	84.57	89.71	88.00	97.71	89.14	95.43	90.76
[49]	91.43	91.43	82.29	92.57	92.57	98.86	91.52
[50]	94.9	90.6	92.0	94.4	95.1	95.1	93.6
[35]	**100.00**	**95.45**	**95.45**	90.63	**100.00**	89.47	**95.17**
[21]	93.14	92.57	93.14	96.57	89.71	**97.71**	93.81
[54]	50.28	79.83	67.14	82.91	60.28	88.51	71.49
Our (SD)	89.52	93.03	91.67	**99.00**	93.21	**97.71**	94.02
Our (SI)	88.44	92.27	91.36	**98.75**	93.53	97.63	93.66

The bold values are the highest classification per class between the benchmark and our proposal.

stage. The difference is not considerable, on the other hand, and with better equipment or code optimization higher FPS can be obtained with higher code complexity, but we deemed it unnecessary given the low classification rate gains for higher complexity than TPOEM$_{5,5,16,1}$.

The full confusion matrix for 7-expression classification using TPOEM$_{5,5,16,1}$ for CK+ is shown in Table 2. The lowest per-class accuracy was 87% (Sadness), which was misclassified 13% of the times as neutral. It is also important to note that some authors [21,27,35,40] report their results using a "modified 7-expression" problem (six basic expressions plus Contempt instead of Neutral). Our approach in such problem for the CK+ database achieved an overall accuracy of 93.0%, whereas the results reported in Lucey et al. [27], Ramirez Rivera et al. [35], Taner Eskil and Benli [40] and Lee et al. [21] were, respectively, 83.3%, 89.3%, 76.8% and 89.6%, respectively.

We also compared the results of TPOEM$_{5,5,16,1}$ for the 6-expression problem using the MMI database with competitive approaches[4], as shown in Table 3. Since these methods report their results for a SD methodology, we also evaluated our method using both SD and SI validation schemes. As can be observed, our SD mean accuracy was inferior only (and by a low margin) than [35], but better than all the other methods. Also, our SI results were only slightly inferior to SD, which indicates good generalization capabilities. It is important to note that the parameters used in the TPOEM codification (W, T, L and P) were selected based on the CK+. They could be optimized for MMI, but our goal here is to show that a set of default parameters can achieve good results in different databases.

To illustrate the potential difference between the results obtained with SD and SI validation schemes for some classifiers, we have also trained an SVM-RBF classifier (used successfully in [35]) with TPOEM features. The results of this classifier (for the 6-expression scenario) for CK+ and MMI using SD and SI validation schemes are shown in Table 4, along with the corresponding results for the proposed classifier. SVM-RBF presented better accuracy in both databases using SD validation, but our method was better when using SI validation. These results indicate a possible overfitting of the SVM-RBF classifier, and also that the proposed classifier seems to generalize better for new subjects in both testes databases.

[4] Results for [45] and [49] were extracted from [21].

Table 4
Comparison of accuracy (%) of classifiers using TPOEM features (our and SVM-RBF) using SD and SI validation schemes.

	Database	
Method	CK+	MMI
Our (SD)	96.92	94.02
Our (SI)	96.87	93.66
SVM-RBF (SD)	99.13	94.72
SVM-RBF (SI)	90.67	90.57

Table 5
Accuracy (%) for the MMI database with a classifier trained using the CK+ database.

Method	Ang.	Dis.	Fea.	Hap.	Sad.	Sur.	Mean
Our	82.4	83.1	79.3	95.6	81.9	92.2	85.8
TPOEM+SVM	48.8	53.2	49.7	70.9	69.0	53.9	57.5
[30]	–	–	–	–	–	–	55.6
[50]	–	–	–	–	–	–	66.9
[29]	–	–	–	–	–	–	56.0

4.3. Cross-dataset validation

Another very important aspect concerning classifiers (particularly in the context of FER) is to evaluate the performance of a classifier trained on one dataset and validated on another one (cross-dataset validation), as noted in [21,22]. In this work, we also used our classifier trained with CK+ images and validated using different datasets.

The first cross-dataset validation uses the MMI database for validation purposed. For each labeled video in the MMI dataset, one frame was chosen as neutral stance and another frame was grouped as the labeled expression. Since L and W parameters are dependent on the size of the image, each detected face was resized to 128×128 pixels, so to match the size of the images used in the training stage. Table 5 shows the average cross-dataset validation results (using CK+ to train and MMI to validate) reported in multiclass-SVM approach presented by Zhang et al. [50], the CNN-based method introduced by Lee et al. [20], and the best combination of classifier and domain adaptation method reported in Miao et al. [29], as well as the per-class and average results produced by our method. As can be observed, the proposed approach presented very good cross-dataset generalization when compared to competitive approaches. For the sake of illustration, we also added the results using TPOEM with a SVM-RBF classifier, as mentioned in Section 4.2. The results were much worse than the proposed classifier, which corroborates the overfitting hypothesis suggested by Table 4.

In fact, one of the main difficulties to obtain good cross-dataset results is the specific condition of the unknown dataset, such as illumination conditions, resolution, individual characteristics of each subject, etc. Classifiers that allow a very flexible decision boundary, such as SVM-RBF or neural networks (including CNNs and deep belief), can actually lead to poor cross-dataset generalization even if the intra-dataset SI validation results are good.

Another experiment used the KDEF (Karolinska Directed Emotional Faces) [28] database. Although KDEF is composed of still images, so that temporal information cannot be explored, it was cho-

Table 6
Accuracy (%) for the KDEF database using our classifier trained with the CK+ database and human evaluation.

	Ang.	Dis.	Fea.	Hap.	Neu.	Sad.	Sur.	Mean
POEM	59.2	79.2	50.0	97.5	97.5	50.0	81.7	73.6
Human	78.8	72.2	43.0	92.6	62.6	76.7	77.1	71.9

sen because it presents FER results inferred by human evaluators, so that a human vs. machine comparison could be performed. We used traditional POEM features in this experiment, more precisely, we used $\text{POEM}_{8,16,3}$ with the proper re-training of the weights w_k presented in Section 3.3.

Table 6 shows our per-class classification results, together with the accuracy obtained by human observers reported by Lundqvist et al. [28]. Results are worse than those obtained for CK+, as expected, but better than human evaluation for five out of seven expressions (and also higher mean accuracy). In fact, KDEF is a challenging database for which even human classification presents large errors, as shown in Lundqvist et al. [28].

5. Conclusions

In this work we presented a classification scheme for the problem of FER using the idea of dynamic texture representation in [52], namely TPOEM. TPOEM codifies the volumetric texture information around a central pixel using spatial and temporal neighbors using directional derivatives and accumulated edge magnitudes along the temporal axis. To account for the increase in code length introduced by using volumetric information (more neighbors), a novel coding scheme was developed aiming to obtain shorter histograms as feature vectors, and at the same time maintain discrimination among volumetric textures. Finally, a simple (and fast) adaptive weighting scheme was explored to combine the TPOEM-based scores at each image patch, producing the final classification result.

The accuracy results using TPOEM features and our simple classifier were comparable to or better than existing algorithms, while keeping computational demands low (when compared to LBP-based methods). It is important to point out that the main results in our work were produced using SI methodologies, which have lower accuracy rates than using conventional 10-folded SD random validation, and can better measure the generalization capability of the classifier (with respect to subjects that were not present in the training stage). In fact, we showed in Table 4 that a same classifier might reach considerably higher accuracy rates when using SD validation when compared do SI, which are misleading and do not represent the actual performance of the classifier.

Our cross-database experiments showed that our the proposed classifier trained with the CK+ database generalizes well for the MMI database, with higher accuracies than competitive approaches. A similar experiment with the KDEF database using POEM-based features indicated that the proposed classifier presented higher mean accuracy than human evaluation.

As future work, we plan to further integrate the TPOEM features with deep learning techniques, which can inherently learn the spatial relationships between the image patches. We also plan to investigate the use of temporal pyramids when computing volumetric textural information, since facial expressions might change at different speeds. Finally, we intend to explore audiovisual FER, by integrating audio into the classifier.

References

[1] T. Ahonen, A. Hadid, M. Pietikäinen, Face recognition with local binary patterns, in: European Conference on Computer Vision, Springer, 2004, pp. 469—–481.
[2] Y.-H. Byeon, K.-C. Kwak, Facial expression recognition using 3d convolutional neural network, Int. J. Adv. Comput. Sci. Appl. 5 (12) (2014).
[3] S.W. Chew, P.J. Lucey, S. Lucey, J.M. Saragih, J.F. Cohn, I.A. Matthews, S. Sridharan, In the pursuit of effective affective computing: The relationship between features and registration, IEEE Trans. Syst., Man, Cybern., Part B 42 (2012) 1006–1016, doi:10.1109/TSMCB.2012.2194485.
[4] H.-C. Choi, S.-y. Oh, Realtime facial expression recognition using active appearance model and multilayer perceptron, in: SICE-ICASE International Joint Conference, IEEE, 2006, pp. 5924–5927.
[5] T.F. Cootes, G.J. Edwards, C.J. Taylor, Active appearance models, IEEE Trans. Pattern Anal. Mach. Intell. 23 (6) (2001) 681–685.
[6] R. Cowie, C. Pelachaud, P. Petta, Emotion-Oriented Systems: The Humaine handbook, 2010, pp. 106–107.
[7] H.-B. Deng, L.-W. Jin, L.-X. Zhen, J.-C. Huang, A new facial expression recognition method based on local gabor filter bank and pca plus lda, Int. J. Inf. Technol. 11 (11) (2005) 86–96.
[8] G. Donato, M. Stewart Bartlett, J.C. Hager, Classifying facial actions, IEEE Trans. Pattern Anal. Mach. Intell. 21 (10) (1999) 974–989.
[9] X. Feng, M. Pietikainen, A. Hadid, Facial expression recognition with local binary patterns and linear programming, Pattern Recogn. Image Anal. C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii 15 (2) (2005) 546–548.
[10] D. Ghimire, J. Lee, Geometric feature-based facial expression recognition in image sequences using multi-class AdaBoost and support vector machines, Sensors 13 (2013) 7714–7734.
[11] S. Happy, A. Routray, Automatic facial expression recognition using features of salient facial patches, IEEE Trans. Affect. Comput. 6 (1) (2015) 1–12.
[12] L.D. Harmon, W.F. Hunt, Automatic recognition of human face profiles, Comput. Graphics Image Process. 6 (2) (1977) 135–156.
[13] E.P. Ijjina, C.K. Mohan, Facial expression recognition using kinect depth sensor and convolutional neural networks, in: Machine Learning and Applications (ICMLA), 2014 13th International Conference on, IEEE, 2014, pp. 392–396.
[14] Y. Ji, K. Idrissi, Automatic facial expression recognition based on spatiotemporal descriptors, Pattern Recognit. Lett. 33 (10) (2012) 1373–1380.
[15] G.J. Kaufman, K.J. Breeding, The automatic recognition of human faces from profile silhouettes, IEEE Trans. Syst., Man Cybern. (1976) 113–121.
[16] M. Kenji, Recognition of facial expression from optical flow, IEICE Trans. Inf. Syst. 74 (10) (1991) 3474–3483.
[17] R.A. Khan, A. Meyer, H. Konik, S. Bouakaz, Human vision inspired framework for facial expressions recognition, in: IEEE International Conference on Image Processing (ICIP), IEEE, 2012, pp. 2593–2596.
[18] B.-K. Kim, J. Roh, S.-Y. Dong, S.-Y. Lee, Hierarchical committee of deep convolutional neural networks for robust facial expression recognition, J. Multimodal User Interfaces 10 (2) (2016) 173–189.
[19] I. Kotsia, I. Pitas, Facial expression recognition in image sequences using geometric deformation features and support vector machines, IEEE Trans. Image Process. 16 (1) (2007) 172–187.
[20] S.H. Lee, W.J. Baddar, Y.M. Ro, Collaborative expression representation using peak expression and intra class variation face images for practical subjec-t-independent emotion recognition in videos, Pattern Recognit. 54 (2016) 52–67.
[21] S.H. Lee, K. Kostas Plataniotis, Y.M. Ro, Intra-Class variation reduction using training expression images for sparse representation based facial expression recognition, IEEE Trans. Affect Comput. 5 (3) (2014) 340–351.
[22] Y. Li, S. Wang, Y. Zhao, Q. Ji, Simultaneous facial feature tracking and facial expression recognition, IEEE Trans. Image Process. 22 (7) (2013) 2559–2573.
[23] P. Liu, S. Han, Z. Meng, Y. Tong, Facial expression recognition via a boosted deep belief network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1805–1812.
[24] A.T. Lopes, E. de Aguiar, A.F. De Souza, T. Oliveira-Santos, Facial expression recognition with convolutional neural networks: coping with few data and the training sample order, Pattern Recognit. 61 (2017) 610–628.
[25] A.T. Lopes, E. de Aguiar, T. Oliveira-Santos, A facial expression recognition system using convolutional networks, in: Graphics, Patterns and Images (SIBGRAPI), 2015 28th SIBGRAPI Conference on, IEEE, 2015, pp. 273–280.
[26] G. Loy, Fast computation of the gabor wavelet transform, in: Digital Image Computing Techniques and Applications, 2002, pp. 279–284.
[27] P. Lucey, J.F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, I. Matthews, The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression, in: IEEE Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, 2010, pp. 94–101.
[28] D. Lundqvist, A. Flykt, A. Öhman, The Karolinska Directed Emotional Faces (KDEF), CD ROM from Department of Clinical Neuroscience, Psychology Section, Karolinska Institutet, 1998, pp. 91–630.
[29] Y.-Q. Miao, R. Araujo, M.S. Kamel, Cross-domain facial expression recognition using supervised kernel mean matching, in: Machine Learning and Applications (ICMLA), 2012 11th International Conference on, 2, IEEE, 2012, pp. 326–332.
[30] A. Mollahosseini, D. Chan, M.H. Mahoor, Going deeper in facial expression recognition using deep neural networks, in: Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on, IEEE, 2016, pp. 1–10.
[31] S. Moore, E. Ong, R. Bowden, Facial expression recognition using spatiotemporal boosted discriminatory classifiers, in: Image Analysis and Recognition, Springer, 2010, pp. 405–414.
[32] M. Pantic, L. Rothkrantz, An expert system for multiple emotional classification of facial expressions, in: IEEE International Conference on Tools with Artificial Intelligence, IEEE, 1999, pp. 113–120.

[33] M. Pantic, M. Valstar, R. Rademaker, L. Maat, Web-based database for facial expression analysis, in: IEEE International Conference on Multimedia and Expo, IEEE, 2005, pp. 5–pp.
[34] M. Quintana, J. Menendez, F. Alvarez, J. Lopez, Improving retail efficiency through sensing technologies: a survey, Pattern Recognit. Lett. 81 (2016) 3–10.
[35] A. Ramirez Rivera, R. Castillo, O. Chae, Local directional number pattern for face analysis: face and expression recognition, IEEE Trans. Image Process. 22 (5) (2013) 1740–1752.
[36] A. Saxena, A. Anand, A. Mukerjee, Robust facial expression recognition using spatially localized geometric model, in: International Conf Systemics, Cybernetics and Informatics (ICSCI), 2004, pp. 124–129.
[37] C. Shan, S. Gong, P.W. McOwan, Robust facial expression recognition using local binary patterns, in: IEEE International Conference on Image Processing, IEEE, 2005, pp. II–370.
[38] C. Shan, S. Gong, P.W. McOwan, Facial expression recognition based on local binary patterns: a comprehensive study, Image Vis. Comput. 27 (6) (2009) 803–816.
[39] E. Silva, C. Esparza, Y. Mejia, POEM-based facial expression recognition, a new approach, in: XVII Symposium ofImage, Signal Processing, and Artificial Vision, IEEE, 2012, pp. 162–167.
[40] M. Taner Eskil, K.S. Benli, Facial expression recognition based on anatomy, Comput. Vision Image Understanding 119 (2014) 1–14.
[41] U. Tariq, J. Yang, T.S. Huang, Supervised super-vector encoding for facial expression recognition, Pattern Recognit. Lett. 46 (2014) 89–95.
[42] V. Terzis, C.N. Moridis, A. Economides, The effect of emotional feedback on behavioral intention to use computer based assessment, Comput. Educ. 59 (2) (2012) 710–721, doi:10.1016/j.compedu.2012.03.003.
[43] N.-S. Vu, A. Caplier, Face recognition with patterns of oriented edge magnitudes, Eur. Conf. Comput. Vis. (2010) 313–326.
[44] N.-S. Vu, A. Caplier, Enhanced patterns of oriented edge magnitudes for face recognition and image matching, IEEE Trans. Image Process. 21 (3) (2012) 1352–1365.
[45] S. Zafeiriou, M. Petrou, Sparse representations for facial expressions recognition via l_1 optimization, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2010, pp. 32–39.
[46] T.H. Zavaschi, A.S. Britto, L.E. Oliveira, A.L. Koerich, Fusion of feature sets and classifiers for facial expression recognition, Expert Syst. Appl. 40 (2) (2013) 646–655.
[47] L. Zelnik-Manor, M. Irani, Event-based analysis of video, in: Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, 2, IEEE, 2001. II–II
[48] Z. Zeng, M. Pantic, G. Roisman, T. Huang, A survey of affect recognition methods: audio, visual, and spontaneous expressions, IEEE Trans. Pattern Anal. Mach. Intell. 31 (1) (2009) 39–58.
[49] S. Zhang, X. Zhao, B. Lei, Robust facial expression recognition via compressive sensing, Sensors 12 (3) (2012) 3747–3761.
[50] X. Zhang, M.H. Mahoor, S.M. Mavadati, Facial expression recognition using l_p-norm mkl multiclass-svm, Mach. Vis. Appl. 26 (4) (2015) 467–483.
[51] Z. Zhang, M. Lyons, M. Schuster, S. Akamatsu, Comparison between geometry-based and Gabor-wavelets-based facial expression recognition using multi-layer perceptron, in: IEEE International Conference on Automatic Face and Gesture Recognition, IEEE, 1998, pp. 454–459.
[52] G. Zhao, M. Pietikäien, Dynamic texture recognition using local binary patterns with an application to facial expressions, IEEE Trans. Pattern Anal. Mach. Intell. 29 (6) (2007) 915–928.
[53] G. Zhao, M. Pietikäinen, Dynamic texture recognition using volume local binary patterns, IEEE Trans. Pattern Anal. Mach. Intell. 29 (6) (2007) 915–928.
[54] L. Zhong, Q. Liu, P. Yang, B. Liu, J. Huang, D.N. Metaxas, Learning active facial patches for expression analysis, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 2562–2569.