Using appearance for improving facial expression recognition

Andrei Zhabinski, Dzmitry Adzinets

Abstract—Traditionally, recognition of a facial expression is performed by extracting and analyzing specific set of landmarks (collectively often referred to as a "shape"). Here we take alternative approach, based not on shapes, but instead on appearance – pixel values within region of interest. We construct classifier based on these data and compare its performance to a shape-based one. We also propose combined method using both sets of features.

Keywords—active appearance models, classification, facial expression recognition, shape and appearance.

I. INTRODUCTION

Understanding emotional state of a person plays key role in such areas as social and psychological research, marketing, gaming industry and many others. In most cases this information is transmitted between people via non-verbal channels. According to [1], about 55% of this information is generated by person's facial expression. Thus, creation of efficient algorithms for expression recognition is mandatory for advanced human-computer interaction.

Existing methods of facial expression recognition vary a lot. Some of them rely on static data (e.g. static images of a face), others - on dynamic ones (e.g. video), some represent input as optical flow, others - as connected vibrations or set of key points. And, of course, algorithms themselves differ a lot. Good overview of different approaches may be found in [2]. In this paper we will concentrate on analysis of static images of a face.

Almost all methods for analyzing static images consist of two steps: feature extraction and recognition itself. Contents of these steps vary a lot, though. For example, on the first step there are 2 main groups of features: general and specific to faces. First group includes all features popular in computer vision. For instance, in [3] Gabor filters are used, while [4] utilizes local binary patterns. Such techniques are easy to implement, but very often resulting features give low accuracy during recognition. Features, specific to faces, include, in particular, action units from FACS coding system [5] and sets of key points, describing shapes of main elements of a face. One method for obtaining key points, known as *active appearance models (AAM)* [6][7][8], gained a lot of attention in the area of facial analysis.

With AAM, recognition is normally performed by extracting coordinates of key points and training classification algorithms on them. Such methods have pretty high accuracy, are intuitive and easily interpretable from psychological point of view. At the same time, this approach ignores information stored in *appearance* - pixel intensities within region of interest. In this paper, we evaluate relative importance of this information and propose combined method that uses both - location of key points and intensity of pixels in the region.

II. SHAPE-BASED METHOD

In order to evaluate importance of appearance for facial expression recognition first of all we need to set up baseline to compare with. In this paper we use a variant of classic algorithm based on position of key points as such baseline.

Essentially, this algorithm consists of 2 steps: obtaining coordinates of key points and classification. Classification of objects by their feature vectors (coordinates of points in this

A. Zhabinski, Belarussian State University of Informatics and Radioelectronics, Minsk, Belarus, (e-mail: andrei.zhabinski@gmail.com).
D. Adzinets, Belarussian State University of Informatics and Radioelectronics, Minsk, Belarus, (e-mail: adzinets@bsuir.by).

case) is a standard task of supervised learning and is of no interest in the context. We only need to mention, that in this paper we used support vector machines for this task, since it demonstrated good results on similar problems.

On other hand, obtaining key points is much more difficult task. According to recent publications (e.g. [2]), it seems like the most popular algorithm for this is active appearance models. Here we describe in short essential steps of this algorithm.

AAM takes a set of images and corresponding sets of key points describing shape of main elements of a face (e.g. see Fig. 1). Using these data algorithm builds 2 statistical models:

- *shape model* - parametric linear model describing possible variations of coordinates of key points. In this context, term "shape" denotes a vector of coordinates of key points: $s = (x_1, y_1, x_2, y_2, \dots, x_n, y_n)$;
- *appearance model* - similar model, but describing possible variations of pixel intensities in the area of interest. Also, unlike shape model, term "appearance" here denotes not a vector, but an image $A(x)$ defined over all pixels $x \in s$.

Fig. 1 Set of key points describing main elements of a face

Note, that here we borrow somewhat inexact, but convenient notation from [7] and write $x \in s$ to refer to all pixels inside shape and not to shape elements themselves.

Since different images may contain different number of pixels inside shape, all images are first aligned to some mean shape. Normally, piecewise affine transformation is used for this: first, set of key points is triangulated, and then every triangle is warped to new coordinates via regular affine transformation (see Fig. 2). Piecewise affine transformation is essential to this work, since it will be also used in appearance-based and (implicitly) in combined methods.

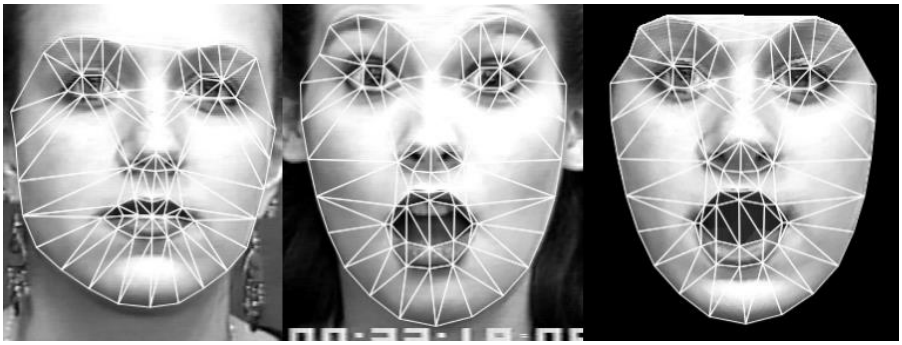

Fig. 2 Example of piecewise affine transformation. On the left: original image and corresponding triangulation. In the middle: target image. On the right: original image warped to the shape of the target one

When applying to new images, algorithm first finds position of a face and applies approximate shape to it. Then, shape and appearance models are iteratively fitted to an image to find exact location of key points. Detailed description of active appearance models can be found in [6] and [7].

Putting it all together we get the following 2 stage algorithm.

Algorithm 1. Shape-based classifier (training)

```
Train AAM: AAM ← aamtrain([x_i for every i])
for each image I_i(x) in dataset do
    Get shape by applying AAM: s_i ←
aamfit(I_i(x))
end
Train classifier: C ← svmtrain([a_i for every
i])
```

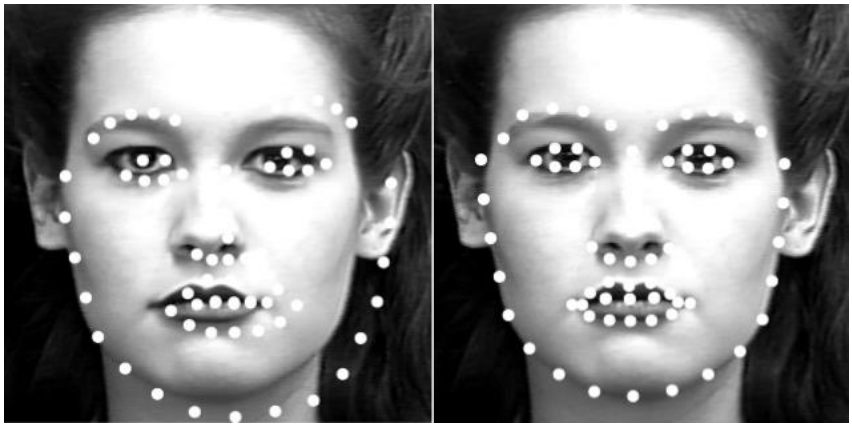

Fig. 3 AAM fitting. On the left: initial guess for a shape. On the right: fitted shape

First, we train AAM on a dataset to use it on later steps. We don't dive deeper into details of AAM training and fitting here because it's not the main topic of this paper, but instead simply state existence of functions *aamtrain* and *aamfit*. Also it's important to note that, though in our experiments we used the same dataset everywhere, dataset for training AAMs should not necessarily be the same as for training main classifier. In fact, it's totally valid to have pre-trained AAM model or even dataset with precomputed coordinates of key points. In our case we obtain these coordinates by fitting trained AAM to every image in the dataset.

Finally, we pass these coordinates to SVM training procedure to get instance of the classifier.

Algorithm 2. Shape-based classifier (predicting)

```
Get new image: I(x) ← new image
Get face shape from it: s ← aamfit(AAM,
I(x))
Apply SVM to get prediction: P ←
svmpredict(C, a)
```

Predicting expression for new image is straightforward: first we obtain that image, then fit AAM to find coordinates of key points and then simply apply trained SVM to get prediction.

Note, that in this method appearance is used purely for obtaining coordinates of key points, but it isn't used on classification step. On the contrary, in next section we will describe method that primarily uses appearance data.

III. APPEARANCE-BASED METHOD

Possibility to learn facial expression from a shape is well known and described in a number of papers. But at the same time this possibility with regard to appearance got from little to no interest in computer vision community. So first of all we need some evidence that appearance holds at least some information about encoded facial expression.

To prove it we propose following informal experiment. We take several images of faces and transform them to the mean shape computed previously using AAMs. We use same piecewise affine transformation which gives us pretty smooth deformation. Result of this transformation is depicted at Fig. 4.

From pictures it's clear that all key points for eyes, eyebrows, nose and mouth have identical coordinates between all images, i.e. all transformed faces have *exactly the same shape*. But it's also noticeable that original facial expressions are still easily guessable. This provides us with evidence that appearance really contains information that encodes human facial expression, and possibly quite a lot of it.

To compare amount of information stored in appearance with one stored in shape, we build another classifier, this time based on pixel intensities.

Algorithm 3. Appearance-based classifier (training)

```
Train AAM: AAM ← aamtrain([x_i for every i])
for each image I_i(x) in dataset do
    Get shape by applying AAM: s_i ← aamfit(I_i(x))
    Warp image to the mean shape: A_i(x) ← W(I_i(x))
    Flatten appearance: a_i ← fl(A_i)
end
Train classifier: C ← svmtrain([a_i for every i])
```

First we preprocess all images in a dataset: fit pre-trained AAM to each of them to obtain shape of a face and then use this shape to warp image to AAM's mean shape. This way all images get same shape (and thus same number of pixels inside), but keep most information about appearance. Then we flatten images, i.e. rearrange pixels inside shape into a single vector of a fixed size. Mathematically, fl denotes a mapping from matrix $A \in R^{m \times n}$ to a vector $a \in R^k$. Finally, we use these flattened appearance vectors to train SVM classifier.

Algorithm 4. Appearance-based classifier (predicting)

```
Get new image: I(x) ← new image
Get face shape from it: s ← aamfit(AAM, I(x))
Warp new image to the mean shape: A(x) ← W(I(x))
Flatten appearance: a ← fl(A)
Apply SVM to get prediction: P ← svmpredict(C, a)
```

During application of classifier to a new image, we follow same steps as during training. First, we apply AAM to obtain shape of a face on the image. Then, using this shape, we warp new image to the mean shape, making it usable for classifier. Finally, we apply trained SVM model to pixels of the transformed image to get prediction.

Before moving further, let's briefly recall common and different parts of two algorithms. Both of them are split into 2 stages - training and predicting. Both require preprocessing to obtain features. First algorithm uses coordinates of key points as its features, while second is based on pixel intensities. In both cases SVM is used as a final classifier.

We should also note, that in this specific work we use AAM for aligning image to the mean shape, though, it's not mandatory precondition. For example, in [9] face is located using several predefined landmarks and then 3D-modeled to obtain possible transformations.

Fig. 4 Transforming images to the mean shape. On the left: images of emotional faces. On the right: same faces transformed to the mean shape. Though shape is the same, facial expressions are still easily distinguishable

IV. COMBINED METHOD

Having 2 similar methods with different sets of features it's natural to combine them in a single algorithm: we can merge information about key point coordinates and pixel intensities into a single vector and use it for classification. Since all essential parts have already been discussed in previous sections, here we will simply list steps of combined algorithm.

Algorithm 5. Combined classifier (training)

```
Train AAM: AAM ← aamtrain([x_i for every i])
for each image I_i(x) in dataset do
    Get shape by applying AAM: s_i ← aamfit(I_i(x))
    Warp image to the mean shape: A_i(x) ← W(I_i(x))
    Flatten appearance: a_i ← fl(A_i)
    Combine vectors: x_i ← [s_i; a_i]
end
Train classifier: C ← svmtrain([a_i for every i])
```

Algorithm is very similar to previous 2 except for the last step in a loop where we combine 2 kinds of features, i.e. simply concatenate their vectors (operator [;] here has signature $[;]: (R^m, R^n) \to R^{m+n}$).

Algorithm 6. Combined classifier (predicting)

```
Get new image: I(x) ← new image
Get face shape from it: s ← aamfit(AAM, I(x))
Warp new image to the mean shape: A(x) ← W(I(x))
Flatten appearance: a ← fl(A)
Combine features: x_i ← [s_i; a_i]
Apply SVM to get prediction: P ← svmpredict(C, a)
```

In the following section we discuss our experiments and results of all 3 described methods.

V. EXPERIMENTS AND RESULTS

We used Extended Cohn-Kanade dataset (CK+)[10] to evaluate our algorithms. This dataset was specifically collected to support research in facial expression tracking and recognition. Currently, it consists of 327 sequences of video frames (from neutral expression to a strongly pronounced emotion) totalling in 10708 images. Each image comes with corresponding shape file, containing coordinates of key points, and every sequence additionally has label of expressed emotion. CK+ uses set of 6 basic emotions proposed by Paul Ekman (e.g. see [11]) and considered the de facto standard for their recognition. These emotions are: happiness, sadness, surprise, fear, disgust and anger.

For training AAM we used AAMToolbox [12], which is freely available for research purposes. Using 4-core CPU Intel i7 and CK+ dataset training AAM took about 3.5 hours. Training SVM classifier on 327 labeled images (only last image from each sequence – the one with maximally expressed emotion – was used) took about 7 seconds.

These timings, however, cover only training stage: during fitting AAM can locate key points within 200 ms, while SVM gives prediction in about 25 ms, which makes it possible to uses our algorithms in near-real time systems.

For evaluating results we used standard method of cross validation. After 10 experiments we got following results:

TABLE 1

ACCURACY OF 3 CLASSIFIERS

algorithm	accuracy
shape-based	89.4%
appearance-based	86.3%
combined	93.5%

As we can see, appearance-based algorithm was able to achieve results pretty close to those of shaped-based one. This means, in particular, that appearance holds pretty large amount of information about facial expression. Also, we see that combined classifier outperformed both of these separate algorithms, which proves that appearance doesn't simply reflect shape variations, but rather adds some amount of unique information.

VI. CONCLUSION AND DISCUSSION

This paper describes and evaluates 3 methods for facial expression recognition, based on shape, appearance and their combination. Results suggest that information about appearance, previously mostly ignored, holds valuable amount of additional information that can be used to significantly improve classification performance. Although here we used raw pixel intensities as appearance features, more sophisticated options are available. In particular, recent progress in deep learning allows obtaining better representation for appearance features. This will become basis for future work.

REFERENCES

[1] Mehrabian, "Communication without words," Psychology Today, vol. 2, no. 4, pp. 53–56, 1968.
[2] S. Z. Li and . K. Jain, Facial expression recognition. London: Springer, 2011.
[3] G. Littlewort, M. S. Bartlett, I. Fasel, J. Susskind, and J. Movellan, "Dynamics of facial expression extracted automatically from video," in J. Image and Vision Computing, pp. 615–625, 2004.
[4] C. Shana, "Facial expression recognition based on local binary patterns: A comprehensive study," Image and Vision Computing, vol. 27, no. 6, pp. 803–816, 2009.
[5] T. Senech al, K. Bailly, and L. Prevost, "Impact of action unit detection in automatic emotion recognition," Pattern Anal. Appl., vol. 17, pp. 51–67, Feb. 2014.
[6] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," in IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 484–498, Springer, 1998.
[7] I. Matthews and S. Baker, "Active appearance models revisited," International Journal of Computer Vision, vol. 60, pp. 135–164, 2003.
[8] M. Ratliff, "Active appearance models for affect recognition using facial expressions," Master's thesis, University of North Carolina, Wilmington, 2010.
[9] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," June 2014.
[10] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression," in Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on, pp. 94–101, IEEE, June 2010.
[11] P. Ekman and E. Rosenberg, What the Face Reveals: Basic and Applied Studies of Spontaneous Expression Using the Facial Action Coding System. Oxford: Oxford University Press, 2005.
[12] "Aamtoolbox." Accessed: 2015-05-22.

This publication is the result of the project implementation:
TEMPUS CERES: Centers of Excellence for young RESearchers.
Reg.no.544137-TEMPUS-1-2013-1-SK-TEMPUS-JPHES