Unsupervised Domain Adaptation for Personalized Facial Emotion Recognition

Gloria Zen, Enver Sangineto
University of Trento
Trento, Italy

Elisa Ricci
FBK Trento, Italy
University of Perugia
Perugia, Italy

Nicu Sebe
University of Trento
Trento, Italy

ABSTRACT

The way in which human beings express emotions depends on their specific personality and cultural background. As a consequence, person independent facial expression classifiers usually fail to accurately recognize emotions which vary between different individuals. On the other hand, training a person-specific classifier for each new user is a time consuming activity which involves collecting hundreds of labeled samples. In this paper we present a personalization approach in which only unlabeled *target-specific* data are required. The method is based on our previous paper [20] in which a regression framework is proposed to learn the relation between the user's specific sample distribution and the parameters of her/his classifier. Once this relation is learned, a target classifier can be constructed using only the new user's sample distribution to transfer the personalized parameters. The novelty of this paper with respect to [20] is the introduction of a new method to represent the source sample distribution based on using only the Support Vectors of the source classifiers. Moreover, we present here a simplified regression framework which achieves the same or even slightly superior experimental results with respect to [20] but it is much easier to reproduce.

Categories and Subject Descriptors

I.2.10 [**Vision and Scene Understanding**]: [Video Analysis]; H.1.2 [**User/Machine Systems**]: [Human factors, Human information processing]

General Terms

Human factors

Keywords

Facial Expression Recognition; Action Unit Detection; Unsupervised Domain Adaptation

ICMI November 12-16th, 2014, Istanbul, Turkey

Figure 1: Sample frames of spontaneous neutral (odd rows) and painful (even rows) facial expressions taken from the PAINFUL dataset. The frames illustrate how human expressions like pain can be exhibited in many different ways, depending on the individual's appearance or personality.

1. INTRODUCTION

There is a long debate in the psychological community on the universality of emotions and on the impact of culture and other environmental aspects on the way human beings express their feelings [8]. Even if the degree of importance of the environment is not yet clear, it is commonly accepted that factors such as the cultural background, gender, age or specific personality traits play a strong role in influencing the modalities and the intensity in which emotions are shown by different people. For instance, the way in which happiness or pain is expressed usually varies, on average, between women and men, introvert and extrovert persons, citizens of different countries, etc. On the other hand, this large variability of behaviors is one of the reasons for which common emotion recognition classifiers usually drop their performance when tested with subjects not belonging to the training set [23, 15]. However, collecting a large number of user-specific labeled data for training a personalized classifier is a time consuming and expensive activity and becomes practically infeasible for easy-to-use Human Computer Interface (HCI) systems.

In this paper we focus on facial expression recognition, which is one of the main modalities in which people express emotions [23] and we show how a user-specific classifier can be built using only unlabeled target data. The method we propose does not make any assumption on the

specific features used to represent the emotions and on the facial expression scenario we adopted in our experiments. As a consequence, it can be applied to different input data types (e.g., audio, accelerometer data, physiological signals, text, etc.) as well as different verbal or non-verbal human communication modalities (e.g. voice recognition, gesture recognition, etc.).

Our idea is based on the framework proposed in [20], where a regression function is learned which associates the (unlabeled) distribution of the user's data in the feature space with the hyperplane parameters of her/his personalized classifier. More specifically, given a set of N *source* users, with their associated *labeled* training samples, a set of corresponding N source linear classifiers (linear SVMs) $C_1, ..., C_N$ is learned. Each classifier C_i is defined by its decision hyperplane $(\mathbf{w}_i, b_i)$. Then, a regression function $f(\cdot)$ is trained which relates the shape of the *unlabeled* data distribution of the i-th source user with her/his specific hyperplane $(\mathbf{w}_i, b_i)$. Once $f(\cdot)$ is learned, labeled data are not necessary any more. Hence, given a new *target* user and her/his *unlabeled* collection of samples, applying $f(\cdot)$ to the new data distribution, the target-specific classifier's parameters can be obtained.

In this paper we build on this idea and we make the following contributions. (1) We simplify the whole framework, e.g., using a scalar Support Vector Regression (SVR) for learning $f(\cdot)$ instead of the Multi-output SVR proposed in [20], which makes the approach easier to be reproduced. (2) More importantly, restricting the chosen classifiers to be linear SVMs, we propose to represent the *source* data distributions by means of the corresponding Support Vectors, which are related with $(\mathbf{w}_i, b_i)$ via the Karush-Kuhn-Tucker (KKT) condition. This makes stronger the geometric relation between $(\mathbf{w}_i, b_i)$ and our non-parametric representation of the source data. We show in Sec. 4 that, using the proposed Support Vector-based representation, we obtain comparable or even better results than [20] on the same benchmarks.

Following [20], we tested our method on different benchmarks and in different application scenarios. In the first scenario we trained a patch-based *facial expression recognition* system based on Local Binary Pattern Histograms (LBPH) [1] and we used the recently proposed PAINFUL dataset [14], which collects videos of real patients with shoulder injuries. The way and the intensity in which the patients spontaneously show pain while performing motion activities largely varies between one person and the other (see Fig. 1). This is witnessed by the experimental results reported in [10], where the authors use the same benchmark and show that person-dependent testing (no training samples of the testing subject used for training) achieves a largely lower performance than a person-independent cross validation. In the second set of experiments, we used the more commonly adopted Extended Cohn Kanade (CK+) face dataset [13] and we trained an *Action Unit detection* system based on facial landmark detection and SIFT extraction. Even if the tasks and the adopted features in the two scenarios are different, we show in Sec. 4 that our approach is able to outperform both a non-personalized classifier and other state-of-the-art unsupervised personalization methods. More importantly, our training strategy is significantly faster (and simpler to implement) than other domain adaptation approaches, most of which are based on time consuming re-training strategies. In Sec. 4 we show that we can compute the target classifier in significantly shorter time, and we believe that this is a important aspect for user personalization in real HCI applications.

2. RELATED WORK

In this section we briefly review the literature related to both single-frame (static) facial expression analysis and domain adaptation/transfer learning techniques.

Single Frame Facial Expression Analysis. Frame-level methods to detect AU occurrence and facial expressions (e.g. happiness, sadness, fear) analyze individual frames [15, 23]. This is typically done first performing face registration, then extracting geometric or appearance features representing each frame, and finally feeding the visual descriptors into static classifiers (e.g. SVM, Boosting, Random Forests). Face registration is commonly achieved through the localization of anatomically salient facial points [19], while in the feature extraction phase both geometric descriptors (e.g. containing information of landmark locations) and appearance features capturing texture changes (e.g. LBPH, SIFT) are typically used.

While state-of-the-art frame-level approaches achieve very accurate recognition in controlled conditions (i.e. frontal face images and posed emotions), the performance significantly degrade in more realistic settings. To address this issue, recent works have focused on recognizing spontaneous facial emotions such as pain [7, 12, 21] and on coping with the challenges of person-specific facial appearance variability [18]. However, most of these works are based on generic detectors, i.e. on classifiers learned with a dataset which is supposed to be representative of all the possible sources of variability (acquisition conditions, differences among individuals, etc.). Unfortunately, having at disposal only datasets of few thousands of images, generalization to arbitrary people appearance and environmental conditions is hard to achieve. In particular, to cope with the issue of facial appearance variability among different people, few recent works have proposed domain adaptation approaches which leverage from unlabeled data depicting the user of interest to learn personalized classifiers. For instance, in [4] an extension of the popular AdaBoost algorithm is proposed for creating user-specific pain recognition models. In [5] an approach based on Kernel Mean Matching [9] named Selective Transfer Machine (STM) is presented for person-specific AU detection. While very accurate recognition results are obtained, STM is very slow at training time. On the other hand to be used in many real world applications, such HCI systems, adaptation algorithms are required to be computationally efficient. Our method is mainly motivated by this need and our experiments (reported in Sec. 4) confirm that it is much faster than [5], being its accuracy comparable or even slightly better.

Transfer learning. Domain adaptation and transfer learning approaches have recently become popular as a means to alleviate the problem of the scarceness of (annotated) training samples for the target domain. In [16] a taxonomy of domain adaptation approaches is presented. According to the type of information which is transferred from the source to the target domain, these methods can be categorized into *instance-transfer*, *features-transfer* and *parameter-transfer* approaches.

Feature-transfer involves finding a shared feature space in which both the source and target data are represented. For instance, in [6] the input features space is augmented originating in a novel descriptor composed of a shared, a source-specific and a target-specific part.

Parameter transfer techniques aim to discover a set of shared parameters or priors between the source and the target models. In Adaptive-SVMs [22] the classifier parameters are obtained training an SVM on the target domain and imposing a regularization term which forces these parameters to be similar to the previously learned source task classifier(s). However, the parameter transfer approaches usually require labeled target data. In our facial expression analysis scenario, this means annotated data for every new user (i.e. users not belonging to the training set) and it is practically infeasible in most applications.

When the target data are unlabeled, instance-transfer approaches are commonly adopted. For instance, in [9] a sample reweighting algorithm is introduced. Gretton et al. propose to compare the centroids of the source and the target distributions and to estimate the weights of the source samples in order to reduce this discrepancy. These weights are then used to assign importance to the source samples when training a model for the target domain. The approach presented in [5] develops from a similar idea. The main drawback of these methods is that computing the difference between the means of the source and the target data distributions in the Reproducing Kernel Hilbert Space may poorly approximate the real distance between distributions. In this paper we circumvent this problem by learning the relation between the "shape" of each user-specific data distribution and the corresponding classifier parameters.

3. REGRESSION-BASED PARAMETER TRANSFER

In this section we present our Support Vector-based Transductive Parameter Transfer (SVTPT) method. This approach is a general framework and it is independent from the types of features and from the applicative scenario. In Sec. 4 we show how this framework can be instantiated with different input modalities (visual descriptors) and classification tasks.

Let $\mathcal{X}, \mathcal{Y}$ be, respectively, a feature space and a label space. In this paper we consider $\mathcal{Y} = \{-1, 1\}$ but generalizing our approach to a multiclass setting is straightforward. We assume that N *labeled source* datasets $D_1^s, ..., D_N^s$, $D_i^s = \{\mathbf{x}_j^s, y_j^s\}_{j=1}^{n_i^s}$, and an *unlabeled target* dataset $X^t = \{\mathbf{x}_j^t\}_{j=1}^{n^t}$, $\mathbf{x}_j^s, \mathbf{x}_j^t \in \mathcal{X}$, $y_j^s \in \mathcal{Y}$, are available. Moreover, let denote $X_i^s = \{\mathbf{x}_j^s\}_{j=1}^{n_i^s}$ the set of points in D_i^s obtained by discarding the label information. D_i^s contains all the training samples specific to the i-th source person, while X^t is the (unlabeled) set of samples of the target individual for whom we aim to construct a personalized classifier.

We assume that the vectors in X_i^s are generated by a marginal distribution P_i^s defined on $\mathcal{X}$ and similarly the vectors X^t are generated by P^t. We generally assume that: $P^t \neq P_i^s$ and $P_i^s \neq P_j^s$ $(1 \leq i, j \leq N,\ i \neq j)$. Finally, we call $\mathcal{P}$ the space of all the possible distributions on $\mathcal{X}$ and we assume that P^t, P_i^s are drawn from $\mathcal{P}$ according to the meta-distribution Π, i.e. $P^t, P_i^s \sim \Pi$ $(1 \leq i \leq N)$.

Our goal is to learn a classifier on the target data X^t without acquiring labeled information. The approach we propose is based on three main phases (Fig. 2). First, a set of source-specific classifiers is learned, one for each training set D_i^s. In the second phase a regression algorithm is adopted in order to learn the relation between the marginal distributions P_i^s and the source classifiers' parameter vectors $\boldsymbol{\theta}_i$. Finally, the desired target classifier $\boldsymbol{\theta}^t$ is obtained by applying the learned distribution-to-classifier mapping and using as input the distribution P^t. In the following, the three phases are described in details.

Source Training Phase 1. In the first phase, each source dataset D_i^s is used to train a classifier solving an optimization problem as:

$$\boldsymbol{\theta}_i = \min_{\boldsymbol{\theta} \in \Theta} \mathcal{R}(\boldsymbol{\theta}) + \lambda_L L(\boldsymbol{\theta}, D_i^s) \tag{1}$$

where $\boldsymbol{\theta}_i$ is the parameter vector associated to the learned classifier, Θ is the parameter space, $\mathcal{R}(\cdot)$ a regularizer and $L(\cdot)$ is the empirical risk weighted with λ_L. Each $\boldsymbol{\theta}_i$ is a personalized classifier (called "ideal" classifier in [5]) because it was trained using the *user specific* data D_i^s. Specifically, in this paper we use a set of linear SVM classifiers, thus $\boldsymbol{\theta}_i = [\mathbf{w}_i',\ b_i]$, $\mathbf{w}_i \in I\!R^M, b_i \in I\!R$, defines a hyperplane in the feature space $\mathcal{X} \equiv I\!R^M$ (see Fig. 2) and it can be estimated by solving:

$$\min_{\mathbf{w}, b} \frac{1}{2}||\mathbf{w}||^2 + \lambda_L \sum_{j=1}^{n_i^s} l(\mathbf{w}'\mathbf{x}_j^s + b, y_j^s) \tag{2}$$

where $l(\cdot)$ is the hinge loss.

Source Training Phase 2. In the second phase, we use a regression approach to learn a mapping $f : \mathcal{P} \rightarrow \Theta$. The intuition here is that each hyperplane, defined by $\boldsymbol{\theta}_i$, depends on the distribution P_i^s generating the data points X_i^s. Thus, if using the source data we are able to learn the relationship between the "shape" of the underlying distribution and its corresponding hyperplane, then, for computing the optimal hyperplane on the target data, we do not need label information any more and we can simply apply the learned mapping $f(\cdot)$, i.e. $\boldsymbol{\theta}^t = f(P^t)$.

Of course, we do not know P_i^s $(1 \leq i \leq N)$ neither P^t, thus we need to approximate both types of distributions using empirical data at disposal. P^t can be approximated using X^t. Similarly, P_i^s can be approximated using X_i^s. However, P_i^s can be approximated also using only the Support Vectors obtained solving (2) for the i-th source classification task. Let $V_i = \{\mathbf{v}_j\}_{j=1}^{m_i}$ be the Support Vectors associated with $\boldsymbol{\theta}_i$ $(V_i \subseteq X_i^s)$. The distribution generating V_i is generally different from the distribution generating X_i^s. In fact V_i does not include those points which are far from the decision hyperplane. Thus approximating P_i^s using V_i introduces an error. Anyway, using V_i instead of X_i^s brings two advantages for our regression framework. The first is that there is a well-known relation between the Support Vectors and the decision hyperplane, given by the KKT condition:

$$\mathbf{w}_i = \sum_{j=1}^{m_i} \alpha_j y_j \mathbf{v}_j, \tag{3}$$

where y_j is the label associated with $\mathbf{v}_j$ and α_j the corresponding Lagrange multiplier obtained solving (2). Note that we do not have labels for the target task, thus (3) cannot be directly used to compute $\mathbf{w}^t$. The KKT condition guarantees the existence of a relation between V_i and $\boldsymbol{\theta}_i$

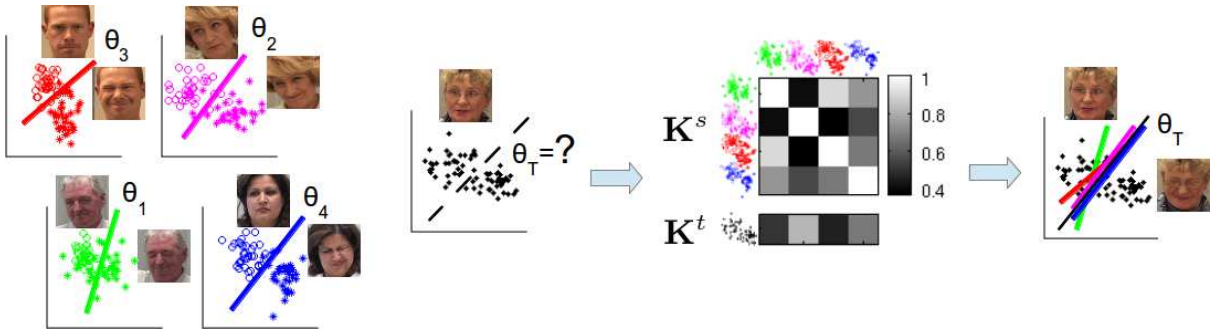

Figure 2: Overview of the training phase. Left: different linear classifiers are learned using labeled source datasets. The goal is to learn a target classifier $\boldsymbol{\theta}^t$ using unlabeled samples. Middle: the source kernel matrix $\mathbf{K}^s$ and the target kernel vector $\mathbf{K}^t$ are computed (see Algorithm 1). Right: $\boldsymbol{\theta}^t$ (black line) is obtained applying the regression function. See the text for more details.

which makes the estimate of our regression function more robust.

The second advantage in using V_i in place of X_i^s to approximate P_i^s is the speed-up and the storage saving we can obtain being usually $m_i < n_i^s$. We show in Sec. 4 that indeed using V_i we obtain slightly more accurate results with a faster training phase.

Let Z_i be the set of points chosen to approximate P_i^s, where either $Z_i = V_i$ or $Z_i = X_i^s$. Given a training set $\mathcal{T} = \{(Z_i, \boldsymbol{\theta}_i)\}_{i=1}^N$ we propose to learn a mapping:

$$\widehat{f} : 2^{\mathcal{X}} \rightarrow \Theta \tag{4}$$

which approximates $f(\cdot)$. The function $\widehat{f}(\cdot)$ is a vector-valued set function, i.e. a function which takes as input a set of points X ($X \subseteq \mathcal{X}$) and outputs a vector $\boldsymbol{\theta} = [\mathbf{w}', b]$. If $\mathcal{X} \equiv I\!R^M$, i.e. $\boldsymbol{\theta}$ is a $M+1$ dimensional vector, $\widehat{f}(\cdot)$ can be learned using $M+1$ independent scalar regressors $\widehat{f}_k(X)$:

$$\widehat{f}(X) = (\widehat{f}_1(X), ..., \widehat{f}_k(X), ..., \widehat{f}_{M+1}(X))' \tag{5}$$

We compute each $\widehat{f}_k(\cdot)$ using the well-known ϵ-insensitive Support Vector Regression framework proposed by Vapnik, which, in our setting, can be formulated as follows. Each $\widehat{f}_k(\cdot)$ ($1 \leq k \leq M+1$) is defined by a set of parameters $\boldsymbol{\pi}_k = (\mathbf{b}_k, c_k)$:

$$\widehat{f}_k(X) = \langle \phi_k(X), \mathbf{b}_k \rangle + c_k, \tag{6}$$

where $\mathbf{b}_k$ and c_k is the weight vector and the bias, respectively, $\phi_k(X)$ is a nonlinear mapping of X to a higher-dimensional space and $\langle X_1, X_2 \rangle$ is a scalar product defined in such a space. In turn $\boldsymbol{\pi}_k$ can be found by minimizing:

$$\min_{\boldsymbol{\pi}} \frac{1}{2}||\boldsymbol{b}_k||^2 + \lambda_E \sum_{i=1}^{N} |\boldsymbol{\theta}_{ik} - \widehat{f}_{\boldsymbol{\pi}}(Z_i)|_\epsilon \tag{7}$$

where $\boldsymbol{\theta}_{ik}$ is the *scalar* value corresponding to the k-th dimension of $\boldsymbol{\theta}_i$, $(Z_i, \boldsymbol{\theta}_i) \in \mathcal{T}$, λ_E is a weight for the empirical risk and $|e|_\epsilon = max(0, |e| - \epsilon)$ is the ϵ-insensitive loss function. The parameter ϵ regulates the accuracy of the approximation.

Eq. (7) can be transformed into the dual problem:

$$\max_{\{\beta_i^k\}} -\frac{1}{2} \sum_{i,l=1}^{N} \beta_i^k \beta_l^k \kappa(Z_i, Z_l) + \sum_{i=1}^{N} \boldsymbol{\theta}_{ik} \beta_i^k - \epsilon \sum_{i=1}^{N} |\beta_i^k| \tag{8}$$

such that:

$$\sum_{i=1}^{N} \beta_i^k = 0, \;\; |\beta_i^k| \leq \lambda_E \;\; (k = 1, ..., M+1). \tag{9}$$

In (8) $\{\beta_i^k\}$ is the set of Lagrange multipliers and $\kappa(Z_i, Z_l) = \langle Z_i, Z_l \rangle$ is the kernel function. Note that the same kernel can be used for all $k = 1, ...M+1$, since $\kappa(Z_i, Z_l)$ estimates the similarity between Z_i and Z_l which is independent from the value of k. In this dual form, (6) becomes:

$$\widehat{f}_k(X) = \sum_{i=1}^{N} \beta_i^k \kappa(Z_i, X) + c_k. \tag{10}$$

From the above formulas, it is clear that both when we compute the parameters of the regression function $\widehat{f}_k(\cdot)$ using (8)-(9) and when this function is applied to a new input using (10), i.e. $\widehat{f}_k(X^t)$, the only thing we need is the kernel function $\kappa(\cdot, \cdot)$, which must be defined for every pair of subsets of $\mathcal{X}$, while the direct definition of the mapping functions $(\phi_k(X), k = 1, ..., M+1)$ is no more necessary.

It is important to note that $\kappa(\cdot, \cdot)$ is defined on *sets* of points and not on feature vectors (i.e., single points) as common kernel functions are. In other words, we need a kernel for estimating the similarity between distributions empirically represented by a set of data points. To this aim we use the non-parametric distribution kernel recently proposed in [2]. This kernel is based on a (single point) Density Estimate kernel and it is defined as follows:

$$\kappa(X_i, X_l) = \frac{1}{nm} \sum_{p=1}^{n} \sum_{q=1}^{m} \kappa_{\mathcal{X}}(\mathbf{x}_p, \mathbf{x}_q), \tag{11}$$

where $X_i = \{\mathbf{x}_p\}_{p=1}^n$, $X_l = \{\mathbf{x}_q\}_{q=1}^m$ and $\kappa_{\mathcal{X}}(\cdot, \cdot)$ is a normalized Gaussian kernel defined on the feature space $\mathcal{X}$.

Target Training Phase. Finally, once $\widehat{f}_k(\cdot)$ is computed using (8)-(9) for every $k = 1, ..., M+1$, in the last phase of our method, we plug (10) into (5) and we apply $\widehat{f}(\cdot)$ to the target data set X^t, obtaining the desired parameter vector of the target classifier:

$$\boldsymbol{\theta}^t = \widehat{f}(X^t). \tag{12}$$

In Algorithm 1 we summarize the whole *training* procedure. Note that the two source training phases need to be computed only once using all the source datasets. Then, for every new target dataset, only the target training phase

Algorithm 1 The proposed SVTPT approach.

Input: The sets $D_1^s, ..., D_N^s$, X^t and the regularization parameters λ_L, λ_E, ϵ.

Source Training Phase 1:
Compute $\{\boldsymbol{\theta}_i = (\mathbf{w}_i, b_i)\}_{i=1}^N$ using (2).
Source Training Phase 2:
Create a training set $\mathcal{T} = \{Z_i, \boldsymbol{\theta}_i\}_{i=1}^N$, where $Z_i = V_i$ or $Z_i = X_i^s$.
Compute the source kernel matrix $\mathbf{K}^s$, $\mathbf{K}_{il}^s = \kappa(Z_i, Z_l)$ using (11)
Given $\mathbf{K}^s$, $\mathcal{T}$, for every $k = 1, ..., M+1$ solve (8)-(9).
Target Training Phase:
Compute the target kernel vector $\mathbf{K}^t$, $\mathbf{K}_i^t = \kappa(Z_i, X^t)$ using (11)
Given $\mathbf{K}^t$, compute $(\mathbf{w}^t, b^t) = \widehat{f}(X^t)$ using (10) and (5).

Output: $\mathbf{w}^t, b^t$

Figure 3: Sample frames of the CK+ dataset. The green rectangles show the cropped part of the image and the dots are the detected facial landmarks (better seen at a high magnification).

needs to be repeated. Thus, when a new target user is introduced into the system, computational times for domain adaptation only depend on the target training phase. Finally, note that all the optimization problems involved in our approach (i.e. (2) and (8)) can be computed with any standard SVM solver. We used the well know LIBSVM library. The whole procedure is shown in Fig. 2.

Testing phase. Once $\boldsymbol{\theta}^t$ has been computed, the test procedure is standard as in classification with linear SVMs. Given a new target feature vector $\mathbf{x}$, the corresponding label y is predicted using: $y = sign((\mathbf{w}^t)'\mathbf{x} + b^t)$.

3.1 Discussion

In this paper we propose to learn the relation between the "shape" of each user-specific data distribution and the corresponding classifier parameters. It is worth noting that, generally speaking, the parameters of a linear classifier (e.g., $\boldsymbol{\theta}_i$ or $\boldsymbol{\theta}^t$) depend not only on the marginal distribution (P_i^s or P^t) but also on the relative distributions of the positives and negatives inside P_i^s and P^t. However, since the classification task is the same for all the users, our assumption is that this relative distributions of positives and negatives inside every user's data is relatively constant and can be learned using source users. We are currently working on extending the experiments with other datasets and our finding is that, as far as data are sufficiently linearly separable, our assumption holds. Representing P_i^s using Support Vectors helps because the classifier parameters are related to the Support Vectors via the KKT condition.

4. EXPERIMENTS

In this section we present two series of experiments to demonstrate the effectiveness of the proposed SVTPT approach in two different applications: AU detection and pain recognition. We deliberately choose these in order to compare SVTPT with our previous work [20] and with the only two other transductive domain adaptation methods for facial expression analysis we are aware of: STM [5] and Transductive AdaBoost (TA) [4]. For the sake of comparison, we adopt the same datasets and features presented in [20, 4, 5]. The performances are evaluated using the F_1 score, defined as $F_1 = \frac{2 \cdot recall \cdot precision}{recall + precision}$, and the Area Under ROC Curve (AUC). Note that the results reported for the other methods we compare with and for our previous work [20] were previously published in [20]. In this paper we are simply restating them for ease of comparison between methods.

4.1 Facial Action Unit Detection

4.1.1 Experimental setup

In this series of experiments we consider the Extended Cohn-Kanade (CK+)[1] dataset [13]. CK+ contains a set of frontal faces depicting spontaneous and posed expressions. The dataset includes 593 videos from 123 users. The number of videos per user ranges from 1 to 11. The video length varies from 4 to 71 frames. Sample frames are shown in Fig. 3.

We followed the same experimental protocol adopted in [5, 20], implementing the same feature extraction pipeline. First, for each frame the face and the facial landmarks are detected, then the face is aligned, cropped and resized to 200×200 pixels. For this pre-processing step, we use the code of [5], available at the author's website[2]. Subsequently, we select 16 landmarks from which we extract SIFT descriptors using OpenCV from 36×36 pixel regions around them. Finally, the descriptors are concatenated and dimensionality is reduced using Principal Component Analysis. We retain 90% of the energy, resulting in feature vectors of size 51. As discussed in [20], we select the most frequent AUs and each AU detection is considered as an independent binary classification problem. Following [5, 20], our experiments are conducted using a leave-one-subject-out evaluation scheme.

4.1.2 Results

We compare our approach with a generic SVM classifier learned on the entire source data, Transductive SVM (TSVM) [11] and domain adaptation-based methods: Kernel Mean Matching (KMM) [9], Domain Adaptation SVM (DASVM) [3] and STM [5]. We also compare with the Transductive Parameter Transfer (TPT) in [20] when the Density Estimate Kernel is used for comparing data distributions. The values concerning the performance of the baseline methods are taken from [5, 20]. The SVTPT parameters (Algorithm 1) have been set with cross-validation. The results are shown in Tables 1 and 2. For the proposed SVTPT method *SVs* means that we use only the support vectors ($Z_i = V_i$) for computing $\mathbf{K}^s$ and $\mathbf{K}^t$ (see Algorithm 1). We also report the results obtained when all the source data points are used, i.e. $Z_i = X_i^s$ (*All*).

[1] http://www.pitt.edu/~emotion/ck-spread.htm
[2] http://humansensing.cs.cmu.edu/intraface/

Table 1: Performance on CK+ dataset. Comparison of different methods based on F_1 Score.

AU	SVM	TSVM [9]	KMM [11]	DASVM [3]	STM [5]	TPT [20]	SVTPT All	SVTPT SVs
1	61.1	56.8	44.9	57.7	74.0	74.4	73.2	74.9
2	73.5	59.8	50.8	64.3	76.2	84.2	81.7	82.4
4	62.7	51.9	52.3	57.7	69.1	66.3	64.5	74.2
6	75.7	47.8	70.1	68.2	79.6	74.8	77.2	74.3
12	76.7	59.6	74.5	59.0	77.2	85.1	83.0	84.6
17	76.0	61.7	53.2	81.4	84.3	76.1	79.0	84.3
Avg	70.9	56.3	57.6	64.7	74.8	76.8	76.7	**79.1**

Table 2: Performance on CK+ dataset. Comparison of different methods based on AUC.

AU	SVM	TSVM [9]	KMM [11]	DASVM [3]	STM [5]	TPT [20]	SVTPT All	SVTPT SVs
1	79.8	69.9	68.9	72.6	88.9	88.2	87.8	89.6
2	90.8	69.3	73.5	71.0	87.5	92.6	92.8	93.9
4	74.8	63.4	62.2	79.9	81.1	84.3	84.5	88.6
6	89.7	60.5	87.7	94.7	94.0	91.7	91.1	91.5
12	88.1	76.0	89.5	95.5	92.8	97.1	96.3	97.5
17	90.3	73.1	66.6	94.7	96.0	94.3	94.4	94.1
Avg	85.6	68.7	74.7	83.1	90.1	91.3	91.3	**92.7**

When support vectors are used for approximating the source data distributions, our approach outperforms all the other methods considering both the AUC and the F_1 score. In particular, SVTPT outperforms the TPT method presented in [20]. This result confirms the intuition that using only Support Vectors for representing the source distributions is more effective than using all the data points (see Sec. 3). In other words, considering only SVs allows to exclude data points which are far away from the hyperplanes and that could be misleading in an unsupervised domain adaptation process. In this sense, the proposed approach comes as an improvement of the method presented in [20]. Among the other baselines, TSVM performs poorly. We ascribe this to the fact that all the source samples are retained during training and the resulting classifier can be wrongly biased when samples from irrelevant individuals are included. KMM, DASVM and STM, instead, re-weight and possibly retain only important data. Among them, STM is the most successful. However, SVTPT and TPT achieve even better performance than STM, probably due to a more effective representation of similarity between data distributions.

4.2 Pain Expression Recognition

4.2.1 Experimental Setup

The PAINFUL[3] dataset is part of the UNBC-McMaster Shoulder Pain Expression Archive Database [14] and is composed of 200 video sequences of 25 patients with shoulder injuries. The videos depicts the patients while performing a series of active and passive range-of-motion tests with either their affected limb or the unaffected one. The facial expressions of the patients are fully spontaneous. The annotation of the video is on a frame basis (48398 frames are labeled by experts using the Prkachin and Solomon Pain Intensity, PSPI, metric system [17]). Some frames are shown in Fig. 4.

To compare our results with [4, 20] we compute the same features. For each frame we use the ground truth eye locations provided in the database to crop and warp the face region into a 128×128 pixel image. The resulting image is divided into 8×8 blocks and LBPH features [1] are computed. We adopt *uniform* $LBP^{u2}_{8,1}$, where *u2* means "uniform" and $(8, 1)$ represents 8 sampling points on a circle of radius 1. For each block a 59-dimensional feature vector is obtained. These vectors are then concatenated resulting into a descriptor of $8 \times 8 \times 59 = 3776$ dimensions. For dimensionality reduction Principal Component Analysis is applied retaining 90% of the variance and obtaining for each frame a feature vector of size 334.

[3]http://www.pitt.edu/∼emotion/um-spread.htm

Figure 4: Sample frames of the PAINFUL dataset. Spontaneous facial expressions of patients under shoulder mobility tests are shown.

Table 3: Performance on PAINFUL dataset, AUC. (*) Results obtained on 30% of data points, see text for details.

AdaBoost [4]	TTA [4]	TSVM [11]	STM* [5]	TPT [20]	SVTPT All	SVTPT SVs
76.9	76.5	69.8	76.8	76.7	76.7	**78.4**

Following [4, 20], our experiments are conducted using a leave-one-subject-out evaluation scheme. However, since there is no pain exhibited in the videos of one subject (i.e. PSPI score is equal to 0 for all the frames), we excluded the videos of this person from the experiments because he cannot be used as a source. Hence, the final number of subjects considered, both at training and at testing time, is 24. Nevertheless, using the non-pain user only as a target, we got slightly better results.

4.2.2 Results

In order to allow a comparison of our results with [4, 20], we evaluate the performance using AUC and we compare against: a generic classifier (AdaBoost) trained using only the source samples (no domain adaptation), Transductive Transfer AdaBoost (TTA) [4], TSVM [11], STM [5] and TPT with Density Estimate Kernel [20]. For TTA and AdaBoost we report the performances published by Chen et al. in [4], and for TPT we report the performances published in [20], while for TSVM and STM we used the codes publicly available[4,5]. Both for TSVM and STM and for TPT and SVTPT, the system's parameters have been set with cross-validation. Table 3 reports the results. As in the case of the CK+ dataset, our approach outperforms all the baseline methods and, concerning SVTPT, a significant improvement in terms of AUC is obtained in the *SVs* case. This confirms again the efficiency of using SVs w.r.t. considering all the data points in our learning from distribution framework, as was initially proposed in [20]. Note that for the case of STM, the results are reported using randomly subsampled 30% of the data points for training. We stopped at this percentage value because the computational time for training a classifier on each subject was over 1 day (see Tab. 6 below) and the time complexity growth is more than linear with respect to the number of training points. Note that this computational

[4]http://svmlight.joachims.org/

[5]http://humansensing.cs.cmu.edu/software.html

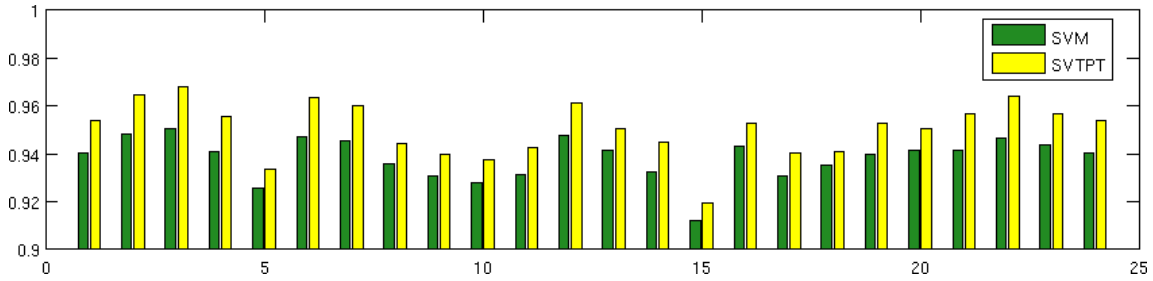

Figure 5: PAINFUL dataset. Similarity of the classifiers θ_i^{SVM} and θ_i^{SVTPT}, with respect to the *personalized* classifiers θ_i, $i = 1, ..., 24$, obtained respectively with SVM and our method (SVTPT) (see the text for details).

Table 4: PAINFUL dataset. (Upper part) Average time cost for training a target classifier for different methods. (Lower part) details on average time costs for different phases of our method.

Method	Data%	Variable to be computed	Training on source	Training on target
SVM	100%	θ^{SVM}	25'	-
TSVM	100%	θ^{TSVM}	-	$1h$ 50'
STM	30%	θ^{STM}	-	$> 1day$
SVTPT All	100%	θ^{SVTPT}	47.1"	0.8"
SVTPT SVs	100%	θ^{SVTPT}	44.1"	0.5"
SVTPT	100%	θ_i; i=1,..,N	37"	-
	100%	$\mathbf{K}^s$ All	10"	-
	100%	$\mathbf{K}^s$ SVs	7"	-
	100%	β_i^k; $1 \leq i \leq N$; $1 \leq k \leq M+1$	0.1"	-
	100%	$\mathbf{K}^t$ All	-	0.8"
	100%	$\mathbf{K}^t$ SVs	-	0.5"

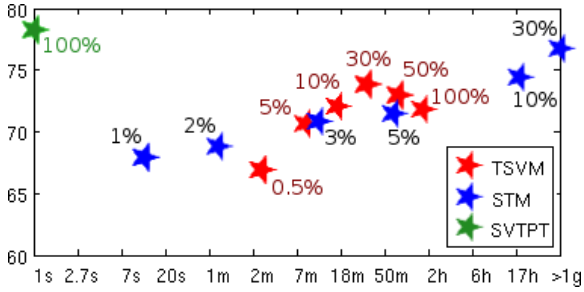

Figure 6: PAINFUL dataset. Performance (AUC) vs average time (in logarithmic scale) for training a target classifier with different unsupervised personalization methods: (red) TSVM [11], (blue) STM [5] and (green) SVTPT. Results for [5, 11] are reported considering different percentage of source data samples. Our method allows the best performance in the shortest time, considering 100% of the data points.

time must be multiplied by 24, the number of leave-one-subject-out cross validation iterations.

In the rest of the experiments, if not otherwise explicitly stated, we refer as SVTPT to indicate our method when training is performed using only Support Vectors (SVs). In order to further validate our approach, in Fig. 5 we show the similarity of the classifiers θ_i^C (where: $C \in \{SVM, SVTPT\}$) with respect to the *personalized* classifiers θ_i for each of the 24 users of the PAINFUL dataset. The personalized classifiers θ_i are trained with labeled data points of only the i-th user (see Eq.(2)). Conversely, each θ_i^{SVTPT} is obtained with the proposed method, as output of our regression function, while θ_i^{SVM} is a generic classifier trained with all and only the data points of the remaining 23 source users. The similarity is computed as $S_i^C = exp(-||\theta_i^C - \theta_i||_2)$, thus the larger its value, the more similar is the classifier to the personalized one. Fig. 5 shows that θ_i^{SVTPT} is closer to θ_i for each of the 24 users of the PAINFUL dataset.

The last part of our experiments is dedicated to compare SVTPT with respect to the state-of-the-art approaches whose code is available on line (i.e. TSVM and STM) in terms of computational times. In Fig. 6 we report the performance versus the target training time cost of SVTPT and the baseline methods. In [4] Chen et al. report the training time for TTA on PAINFUL (17.6 minutes) but they do not mention the workstation they used, thus the results are not directly comparable. Our experiments were run on a 4 Cores 2.40GHz CPU machine. It is clear from this plot that both TSVM and STM do not scale well as the number of training samples increases. For instance, using only 10% of the source samples, TSVM needs 17 minutes on average for training a personalized classifier, STM 18 hours and SVTPT only less than a second. Even if the accuracy of all the methods are comparable in this challenging dataset, our proposed approach largely outperforms all the other algorithms with respect to the computational cost. We believe that time efficiency is crucial in real world multimedia applications where the construction of a personalized classifier needs to be fast to attract real users.

In Table 4 (upper part) we report the computational costs for the different methods during the two phases: (i) *source training*, which involves only source data, and (ii) *target training*, in which also target data are used. In Table 4 (lower part) we report the computational costs of the main SVTPT operations (see Algorithm 1). SVTPT is the only method whose computational cost is split between these two phases, and the most expensive operations (i.e. computation of $\mathbf{K}^s$ and θ_i) are executed only once, as they are target independent. Note that, in a HCI setting, the computational cost of the second phase is the only one perceived by a new user. Conversely, other personalization methods, such as TSVM and STM are based on *training* a new classifier for each target user, an operation which is generally more costly then applying a pre-learned regression function. Moreover, the STM iterative optimization steps do not scale well with large datasets. Finally, the generic SVM classifier does not have any computational cost at target training time, as no personalization is performed. However, training the generic SVM is more costly then SVTPT, taking about 25 minutes

versus less than 1 minute. The reason is that the computational complexity of the SVM solvers varies between $O(m^2)$ and $O(m^3)$, with m the number of training data points. In the case of PAINFUL dataset, training 1 classifier over $\sim 45k$ data samples is more costly than training N source classifiers, with an average of $\sim 1.9k$ data samples each.

4.3 Conclusions

In this paper we proposed a novel domain adaptation approach for facial expression analysis which deals with the inter-individual variability of expressing emotions by learning a set of personalized facial expression classifiers. With our approach a classifier for a new target individual is inferred without the need of acquiring labeled data. The proposed method relies on a regression framework to learn a mapping between a marginal distribution of the datapoints associated to a given person and the parameters of her/his personalized classifier. This distribution is represented by the set of Support Vectors of the linear classifier in the source case and by all the unlabeled data points in the target case. While we tested our method on facial analysis problems (AU detection and pain recognition), its formulation is completely general and potentially applicable to many other (visual or non-visual) domain adaptation problems. We showed that our system achieves state-of-the-art accuracy on public benchmarks while being different orders of magnitude faster than other unsupervised domain adaptation approaches.

5. ACKNOWLEDGMENTS

This work was partially supported by the European 7th Framework Program, under grant VENTURI (FP7-288238) and the Active Ageing at Home cluster project.

6. REFERENCES

[1] T. Ahonen, A. Hadid, and M. Pietikäinen. Face description with local binary patterns: Application to face recognition. *IEEE Trans. on PAMI*, 28(12):2037–2041, 2006.

[2] G. Blanchard, G. Lee, and C. Scott. Generalizing from several related classification tasks to a new unlabeled sample. In *NIPS*, 2011.

[3] L. Bruzzone and M. Marconcini. Domain adaptation problems: A DASVM classification technique and a circular validation strategy. *IEEE Trans. on PAMI*, 32(5):770–787, 2010.

[4] J. Chen, X. Liu, P. Tu, and A. Aragones. Learning person-specific models for facial expression and action unit recognition. *Pattern Recognition Letters*, 34(15):1964–1970, 2013.

[5] W.-S. Chu, F. De La Torre, and J. F. Cohn. Selective transfer machine for personalized facial action unit detection. In *CVPR*, 2013.

[6] H. Daume. Frustratingly easy domain adaptation. In *Proc. of Association for Computational Linguistics*, pages 256–263, 2007.

[7] H. Dibeklioğlu, T. Gevers, A. A. Salah, and R. Valenti. A smile can reveal your age: Enabling facial dynamics in age estimation. In *ACM Multimedia*, 2012.

[8] P. Ekman. Universals and cultural differences in facial expressions of emotion. In *Proc. Nebraska Symp. Motivation*, 1971.

[9] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf. Covariate shift by kernel mean matching. *Dataset shift in machine learning*, 3(4):5, 2009.

[10] Z. Hammal and J. F. Cohn. Automatic detection of pain intensity. In *ICMI*, 2012.

[11] T. Joachims. Transductive inference for text classification using support vector machines. In *ICML*, 1999.

[12] G. Littlewort, M. S. Bartlett, and K. Lee. Faces of pain: automated measurement of spontaneous facial expressions of genuine and posed pain. In *ICMI*, 2007.

[13] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression. In *CVPR Workshops (CVPRW)*, 2010.

[14] P. Lucey, J. F. Cohn, K. M. Prkachin, P. E. Solomon, S. W. Chew, and I. Matthews. Painful monitoring: Automatic pain monitoring using the unbc-mcmaster shoulder pain expression archive database. *Image Vision Comput.*, 30(3):197–205, 2012.

[15] A. Martinez and S. Du. A model of the perception of facial expressions of emotion by humans: Research overview and perspectives. *Journal of Machine Learning Research*, 13:1589–1608, 2012.

[16] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Trans. on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[17] K. M. Prkachin and P. E. Solomon. The structure, reliability and validity of pain expression: Evidence from patients with shoulder pain. *Pain*, 139(2):267–274, 2008.

[18] O. Rudovic, V. Pavlovic, and M. Pantic. Context-sensitive conditional ordinal random fields for facial action intensity estimation. In *ICCV Workshops*, 2013.

[19] E. Sangineto. Pose and expression independent facial landmark localization using dense SURF and the Hausdorff distance. *IEEE Trans. on PAMI*, 35(3):624–638, 2013.

[20] E. Sangineto, G. Zen, E. Ricci, and N. Sebe. We are not all equal: Personalizing models for facial expression analysis with transductive parameter transfer. In *ACM Multimedia*, 2014.

[21] M. F. Valstar, M. Pantic, Z. Ambadar, and J. F. Cohn. Spontaneous vs. posed facial behavior: automatic analysis of brow actions. In *ICMI*, 2006.

[22] J. Yang, R. Yan, and A. G. Hauptmann. Adapting svm classifiers to data with shifted distributions. In *IEEE International Conference on Data Mining (ICDM) Workshops*, 2007.

[23] Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *IEEE Trans. on PAMI*, 31(1):39–58, 2009.