Doubly Sparse Relevance Vector Machine for Continuous Facial Behavior Estimation

Sebastian Kaltwang, Sinisa Todorovic, *Member, IEEE* and Maja Pantic, *Fellow, IEEE*

Abstract—Certain inner feelings and physiological states like pain are subjective states that cannot be directly measured, but can be estimated from spontaneous facial expressions. Since they are typically characterized by subtle movements of facial parts, analysis of the facial details is required. To this end, we formulate a new regression method for continuous estimation of the intensity of facial behavior interpretation, called Doubly Sparse Relevance Vector Machine (DSRVM). DSRVM enforces double sparsity by jointly selecting the most relevant training examples (a.k.a. relevance vectors) and the most important kernels associated with facial parts relevant for interpretation of observed facial expressions. This advances prior work on multi-kernel learning, where sparsity of relevant kernels is typically ignored. Empirical evaluation on challenging Shoulder Pain videos, and the benchmark DISFA and SEMAINE datasets demonstrate that DSRVM outperforms competing approaches with a multi-fold reduction of running times in training and testing.

Index Terms—Regression, Relevance Vector Machine, Multiple Kernel Learning, Facial expressions

1 Introduction

Spontaneous facial expressions are a window to our inner feelings and thoughts. They communicate emotions, clarify and stress what is being said, and signal comprehension, disagreement and stances [9]. It is not surprising then that machine understanding of human facial expressions could revolutionise the way we interact with computers, robots and cars; such technology would enable these artifacts to react properly when their users are tired, stressed and bored. Hence, machine understanding of facial expressions has recently become a hot research topic.

Most work to date focused on detection of the presence or absence of a certain facial expression (e.g. prototypic expression of happiness) or of a certain facial action (e.g. a smile, which is coded as AU12 in FACS [8]), instead on their full range intensity estimation [32], [54]. Yet, the meaning and function of spontaneous facial expressions depends largely on their intensity. For example, the smiles of enjoyment are full-blown smiles, while the "fake happiness smiles" (as in sarcasm) may be asymmetric and are usually less in intensity when observed in naturalistic social settings. As noted in [13], "most of the smile genuineness impression is created by the intensity of the smile".

Furthermore, most of the past work on the topic treats the observed facial region holistically rather than a sum of its part [32], [54]. Yet findings from psychological research suggest that the brain processes facial expressions as a set of its parts (cf. facial actions) rather than holistically [1]. This also forms the basis of componential facial emotion theory, which suggests that only components of facial expressions (facial actions) are universally displayed, and that only components of expressions play a role in facial expression interpretation, not full expressions [31]. This explains why humans can 'fill in' the missing parts of an occluded facial expression and judge expressed emotional states even though just some facial actions are visible.

In contrast to earlier work in machine understanding of facial expressions, we study spontaneous facial behavior in video for identifying the intensity levels of:

1) Facial Action Units (FAU) of the facial action coding system (FACS) [8],
2) Two factors of emotional experience: Valence – how much negative or positive, and Arousal – how much calming or exciting is the experience, and
3) Shoulder pain of patients during arm movement tests.

Since our goal is to identify the intensity of (1)–(3), our problem is that of continuous estimation of spontaneous facial behavior. This problem is challenging for a number of reasons. In general, spontaneous facial expressions are characterized by subtle facial deformations that are difficult to track, and frequent out-of-plane head movements whose effects are difficult to remove. By patients with pain, considered in this paper, facial expressions are typically subdued, due to a long-term exposure to pain. Moreover, near-by intensity levels of emotional experience (or pain) are typically manifested by very small differences in facial expressions. All these challenges require a *fine-grained* approach capable of identifying the most relevant facial details and their subtle movements.

1.1 Motivation

Psychological studies on facial expressions agree on two key findings that motivate our approach.

- *S. Kaltwang and M. Pantic are with the Department of Computing, Imperial College London, 180 Queens Gate, London SW7 2AZ, UK. M. Pantic is also with the EEMCS, University of Twente, the Netherlands. E-mail: {sk2608, m.pantic}@imperial.ac.uk.*
- *S. Todorovic is with the School of EECS, Oregon State University, 1148 Kelley Engineering Center, Corvallis, OR 97331. E-mail: sinisa@eecs.oregonstate.edu.*

First, our motivation to estimate the intensity of valance, arousal, and pain by analyzing facial behavior on a *continuous* scale rather than in terms of discrete levels stems directly from the relevant research in psychology [9], [15], [16]. That research has found that the intensity of spontaneous facial expressions are proportional to the intensity of underlying affective states, modulated by a particular social situation. For example, the vigor of spontaneous eye squints and brow scowls reveals the intensity of the felt pain [5]. Consequently, the *continuous-valued* intensity of people's affective and physiological states (e.g. pain) – which cannot be directly measured – can be effectively estimated from *continuous* facial behavior estimation. In fact, there are many limitations and biases of verbal self-reports, and great benefits of measures based on nonverbal facial behavior [4]. E.g., this is currently the most prominent line of research in psychological and clinical studies of pain [5], [52]. This also explains why machine understanding of pain intensity from facial expressions would be beneficial in those studies.

Second, we propose here a method for automated FACS coding of shown facial expressions. FACS defines 32 FAUs, considered to be the smallest visually discernible facial movements directly related to contractions of the underlying muscles. FACS has been developed for human observers and it provides the rules for the recognition of these 32 FAUs and their intensity, which is defined using the five level FACS model (A<B<C<D<E) [8]. This 5-level FACS model for FAU intensity scoring is useful for human annotators who then do not have to depict finer differences between the intensity of the observed facial movements. But, on the other hand, this model is too crude for computer-vision-based approaches that can easily track and estimate very fine differences in the magnitude of the tracked motion (e.g. see [50]). Hence, in this work, we approach the problem of FACS coding as a continuos-value estimation problem.

1.2 Overview of Our Contributions

We cast our continuous estimation problem within the regression framework, and formulate a new regressor – called Doubly Sparse Relevance Vector Machine (DSRVM). DSRVM identifies the most relevant training examples of face snapshots – termed relevance vectors – which improve regression. Simultaneously, DSRVM also identifies the most informative parts of relevant training faces. To this end, DSRVM uses a bank of kernel functions and the selection of informative facial parts is formalized as a selection of optimal kernel functions from the bank. To avoid overfitting, and reduce computation complexity, we regularize DSRVM to be twofold sparse in terms of both relevance vectors and kernels.

DSRVM simultaneously learns multiple kernels within a probabilistic framework. This allows *computationally efficient* EM learning and *doubly sparse* solutions, where the learned DSRVM uses only a few kernels and a few relevance vectors. This advances related multiple-kernel learning (MKL) methods [11], [34], [35], [46]. They are typically specified within the max-margin framework, where enforcing sparsity in both primal and dual domains is computationally intractable, and thus requires approximations [58]. The existing MKL methods enforce sparsity only by selecting a few relevance vectors; however, the resulting number of relevant kernels can be prohibitively large.

We present empirical evaluation on challenging videos of the benchmark Shoulder Pain [24], SEMAINE [26], and DISFA [25] datasets. The experiments demonstrate many advantages of DSRVM, in comparison with competing approaches, in terms of higher accuracy and reduced computation complexity.

In the sequel, Sec. 2 reviews prior work; Sec. 3 formalizes DSRVM; Sec. 4 explains our differences from RVM; Sec. 5 presents our differences from related MKL methods; Sec. 6 and Sec. 7 specify video features and kernels that we use for DSRVM regression; Sec. 8 describes the four datasets we use for evaluation and shows our experimental results; and Sec. 9 presents our concluding remarks.

2 Related Prior Work

This section reviews prior work on: machine analysis of facial expressions in a continuous domain, pain intensity estimation and FAU-intensity estimation. A more detailed explanation of our differences from Relevance Vector Machine [47], [48], and existing MKL methods — namely, SimpleMKL (SMKL) [35], and multi-kernel RVM [3], [6] — is deferred to Sec. 4 and Sec. 5.

In comparison to continuous-domain affect recognition from speech, continuous-based analysis of facial signals for affective computing is relatively underexplored research direction. A few existing approaches are narrowly aimed at inferring either valence and arousal intensities [12], or pain intensity [19] based on holistic dynamics of the appearance and characteristic points of a face as a whole.

Estimation of pain from facial expressions has been typically cast as a binary classification problem [10], [21], [23], [27], i.e., that of recognizing pain vs. no-pain, or as a multiclass problem [14], [38], i.e., that of recognizing a few ordinal levels of pain intensity. Except our previous work [19], we are not aware of any other approach to continuous-domain pain intensity estimation from facial behavior.

Our approach performs fine-grained continuous-value estimation of spontaneous facial expressions at every video frame. In contrast, most works on FAU-intensity modeling use the three point ordinal scale – namely, onset-apex-offset – corresponding to the three characteristic temporal segments of the facial behavior [18], [41], [51]. Only a few existing approaches estimate FAU intensities for each video frame [17], [25], [38], [42]. However, they use holistic appearance features, extracted from the entire face, and do not account for relevant facial parts.

Most approaches to automatic facial behavior estimation typically analyze the face as a whole [54]. They usually estimate temporal changes of facial appearance or facial feature points extracted from the entire face (e.g. [19], [23], [51]). The only exceptions include the part-based

methods for detecting facial actions units (FAUs) [20], [41], [56], those for classifying basic emotion categories [22], [53], [57], and those for pain classification [21], [27]. However, these approaches are not suitable for our problem due to the following limitations. Except for [19], none of the works performs intensity estimation. The methods of [20] and [41] identify important facial parts for detecting FAUs, but they do not account for interactions between the parts. Consequently, they underperform in the case when two (or more) FAUs simultaneously co-occur — which is quite frequent in spontaneous facial expressions — since this modifies the appearance of facial parts relative to single FAU occurrences. Also, the work of [53] seems inappropriate for our purposes, because of its poor trade-off between complexity and accuracy. It uses a computationally expensive graph matching for identifying relevant facial parts and their relationships for emotion categorization. [57] is a stage-wise approach, which first selects the patches using Multi-task sparse learning and then classifies those using SVM. [22] combines sparse linear SVM with multi-task learning for recognizing the six basic emotions. They learn sparse emotion-specific and shared sets of feature dimensions. However, the number of selected dimensions is non-adaptive and needs to be pre-defined. [56] jointly detects multiple FAUs, while selecting a sparse set of facial patches and adhering to pre-defined FAU co-occurrences. In contrast to our approach, the method is limited to classification and to a linear prediction function in the feature space. Finally, the methods in [21], [27] select features for pain classification. However, the selection is done independently as a preprocessing step.

3 THE MODEL

This section specifies our DSRVM which is aimed at the following regression problem. Suppose we are given training video frames showing spontaneous facial expressions, $\mathcal{D} = \{(\mathbf{x}_n, t_n)\}_{n=1}^N$, where $\mathbf{x}_n$ is a feature vector, and t_n is the associated target value corresponding to the intensity level of a person's emotional experience (e.g., real-valued valence or arousal). Our goal is to find a function, y, that models $y(\mathbf{x}) = t$ for any $(\mathbf{x}, t)$ pair.

For regression, DSRVM uses a weighted sum of M basis functions, $y(\mathbf{x}) = \sum_{m=1}^M w_m \phi_m(\mathbf{x})$, where $\mathbf{w} = [w_1 ... w_M]^\top$ weight the contribution of basis functions $\{\phi_m\}_{m=1}^M$ in the sum. The mth basis is computed by centering a kernel $\boldsymbol{\kappa}$ at the mth training data point, $\phi_m(\mathbf{x}) = \boldsymbol{\kappa}(\mathbf{x}, \mathbf{x}_m)$ and $M = N$. The kernel $\boldsymbol{\kappa}$ is defined as

$$\boldsymbol{\kappa} = \textstyle\sum_{k=1}^K v_k \kappa_k, \tag{1}$$

where $\{\kappa_k\}_{k=1}^K$ is a set of predefined kernels, and $\mathbf{v} = [v_1 ... v_K]^\top$ are their corresponding weights. κ_k could be any kernel function, like Radial Basis Function (RBF), and histogram intersection kernels. There is no restriction to Mercer kernels, as in Support Vector Machines [45].

Thus, DSRVM defines the regression function y as

$$y(\mathbf{x}; \mathbf{w}, \mathbf{v}) = \textstyle\sum_{m=1}^M \sum_{k=1}^K w_m v_k \kappa_k(\mathbf{x}, \mathbf{x}_m). \tag{2}$$

DSRVM is a *doubly sparse* model, because learning seeks to identify a small subset of non-zero weights w_m and v_k, whereas the remaining weights are zero. This means that a sparse set of basis functions ϕ_m will be used for regression. Since each ϕ_m is associated with $\mathbf{x}_m$, the training data with nonzero weights in (2) are called the relevance vectors (RV). Following this convention, the κ_k with non-zero weights are called the *relevance kernels* (RK).

DSRVM solves the regression model defined by (2) in a Bayesian way, and therefore the next step is to define probability distributions for the error and the parameters of (2). We assume an additive Gaussian error ϵ with zero mean and variance σ^2, i.e. $t = y(\mathbf{x}; \mathbf{w}, \mathbf{v}) + \epsilon$ and thus:

$$p(t|\mathbf{w}, \mathbf{v}, \sigma^2, \mathbf{x}) \sim \mathcal{N}\left(t; y(\mathbf{x}; \mathbf{w}, \mathbf{v}), \sigma^2\right), \tag{3}$$

where $\mathcal{N}\left(t; y, \sigma^2\right)$ denotes the Gaussian distribution over the variable t with mean y and variance σ^2. Note that (3) holds for all training $(\mathbf{x}_n, t_n)$ pairs as well as the test data $(\mathbf{x}_{\text{new}}, t_{\text{new}})$. Furthermore, we assume independence between the observations, i.e. $p(\mathbf{t}|\mathbf{w}, \mathbf{v}, \sigma^2, \mathbf{X}) = \prod_n p(t_n|\mathbf{w}, \mathbf{v}, \sigma^2, \mathbf{x}_n)$, where $\mathbf{t} = [t_1, ... t_N]^\top$ and $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]^\top$. In order to enforce sparse weights, we define a hierarchical Gaussian prior over $\mathbf{w}$ and $\mathbf{v}$:

$$p(\mathbf{w}|\boldsymbol{\alpha}) \sim \mathcal{N}(\mathbf{w}; \mathbf{0}, \mathbf{A}^{-1}) \tag{4}$$

$$p(\mathbf{v}|\boldsymbol{\beta}) \sim \mathcal{N}(\mathbf{v}; \mathbf{0}, \mathbf{B}^{-1}) \tag{5}$$

with the hyper-parameters $\mathbf{A} = \text{diag}(\boldsymbol{\alpha})$, $\boldsymbol{\alpha} = [\alpha_1, ..., \alpha_M]^\top$ and $\mathbf{B} = \text{diag}(\boldsymbol{\beta})$, $\boldsymbol{\beta} = [\beta_1, ..., \beta_K]^\top$. Furthermore, we assume a uniform prior distribution for the hyper-parameters. When integrated out, the hierarchical prior leads to the improper sparse prior over $\mathbf{w}$ and $\mathbf{v}$ with $p(w_m) \sim 1/|w_m|$ (analogous for v_k) , which is similar to the sparse Laplace distribution [47]. A plates diagram of the model is depicted in Fig. 1. A full Bayesian treatment

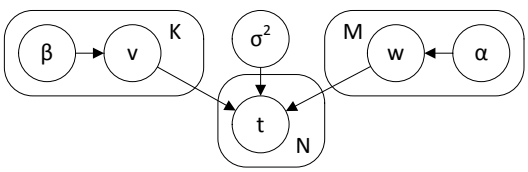

Fig. 1: Plates diagram of the model.

of the model would lead to the predictive distribution for a new target t_{new}, given the features $\mathbf{x}_{\text{new}}$:

$$p(t_{\text{new}}|\mathbf{t}, \mathbf{X}, \mathbf{x}_{\text{new}}) = \textstyle\int p(t_{\text{new}}|\boldsymbol{\Omega}, \mathbf{x}_{\text{new}}) p(\boldsymbol{\Omega}|\mathbf{t}, \mathbf{X}) d\boldsymbol{\Omega} \tag{6}$$

where $\boldsymbol{\Omega} = (\mathbf{w}, \mathbf{v}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2)$ is the set of all parameters. Hence, the training procedure needs to find the posterior distribution $p(\boldsymbol{\Omega}|\mathbf{t}, \mathbf{X})$. Since this posterior is intractable without further assumptions, we employ a type-II maximum likelihood (ML) estimate of the hyper-parameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$ and a ML estimate for σ^2. To improve readability, we leave out the conditioning on $\mathbf{X}$ in the following. The posterior composes into $p(\boldsymbol{\Omega}|\mathbf{t}) = p(\mathbf{w}, \mathbf{v}|\mathbf{t}, \boldsymbol{\alpha}_*, \boldsymbol{\beta}_*, \sigma_*^2) p(\boldsymbol{\alpha}_*, \boldsymbol{\beta}_*, \sigma_*^2|\mathbf{t})$, where $\boldsymbol{\alpha}_*, \boldsymbol{\beta}_*, \sigma_*^2$ are the ML estimates of the corresponding parameters. The joint posterior of the weight parameters $(\mathbf{w}, \mathbf{v})$ cannot be explicitly calculated, and hence it is

approximated by a variational distribution that factorizes with respect to $\mathbf{w}$ and $\mathbf{v}$:

$$p(\mathbf{w}, \mathbf{v}|\mathbf{t}, \boldsymbol{\alpha}_*, \boldsymbol{\beta}_*, \sigma_*^2) \approx q(\mathbf{w})q(\mathbf{v}), \tag{7}$$

where the factors $q(\mathbf{w})$ and $q(\mathbf{v})$ are arbitrary distributions whose explicit form is derived as follows. Since t does not depend on $\boldsymbol{\alpha}_*$ and $\boldsymbol{\beta}_*$ if the posterior of $\mathbf{w}$ and $\mathbf{v}$ is given, the predictive distribution (6) can be approximated by

$$p(t_{\text{new}}|\mathbf{t}, \mathbf{x}_{\text{new}}) \approx \int p(t_{\text{new}}|\mathbf{w}, \mathbf{v}, \sigma_*^2, \mathbf{x}_{\text{new}})q(\mathbf{w})q(\mathbf{v})d\mathbf{w}d\mathbf{v} \tag{8}$$

and the variational lower bound of the marginal log-likelihood $p(\mathbf{t}|\boldsymbol{\alpha}_*, \boldsymbol{\beta}_*, \sigma_*^2)$ is

$$\mathcal{L} = \int q(\mathbf{w})q(\mathbf{v}) \log \left(\frac{p(\mathbf{w}, \mathbf{v}, \mathbf{t}|\boldsymbol{\alpha}_*, \boldsymbol{\beta}_*, \sigma_*^2)}{q(\mathbf{w})q(\mathbf{v})} \right) d\mathbf{w}d\mathbf{v}. \tag{9}$$

The DSRVM training algorithm maximizes the approximated log-likelihood $\mathcal{L}$ by repeating 5 update steps, which use a mix of Variational Inference and Expectation-Maximization. $q(\mathbf{w})$ and $q(\mathbf{v})$ are derived by variational methods and the parameters $\boldsymbol{\alpha}_*$, $\boldsymbol{\beta}_*$ and σ_*^2 maximize $\mathcal{L}$ given the expectations of $q(\mathbf{w})$ and $q(\mathbf{v})$. Further derivation details are provided in the supplementary material, Appendix B. The resulting update steps are specified as:

Step 1: Re-estimate $q(\mathbf{w})$

$$\begin{aligned} q^\star(\mathbf{w}) &\sim \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma}), \\ \boldsymbol{\mu} &= \sigma^{-2}\boldsymbol{\Sigma} \left(\textstyle\sum_k \mathbb{E}\left[v_k\right] \boldsymbol{\Phi}_k^\top \right) \mathbf{t}, \\ \boldsymbol{\Sigma} &= \left(\mathbf{A} + \sigma^{-2} \textstyle\sum_{k_1} \sum_{k_2} \mathbb{E}\left[v_{k_1} v_{k_2}\right] \boldsymbol{\Phi}_{k_1}^\top \boldsymbol{\Phi}_{k_2} \right)^{-1}, \end{aligned} \tag{10}$$

where $\mathbb{E}[\cdot]$ is the expected value and $\boldsymbol{\Phi}_k \in \mathbb{R}^{N \times M}$ is the kth matrix slice along the 3rd dimension of the kernel design tensor $\boldsymbol{\mathcal{K}} \in \mathbb{R}^{N \times M \times K}$ with $\boldsymbol{\mathcal{K}}(n, m, k) = \kappa_k(\mathbf{x}_n, \mathbf{x}_m)$.

Step 2: Re-estimate $q(\mathbf{v})$

$$\begin{aligned} q^\star(\mathbf{v}) &\sim \mathcal{N}(\mathbf{v}; \boldsymbol{\nu}, \boldsymbol{\Lambda}), \\ \boldsymbol{\nu} &= \sigma^{-2}\boldsymbol{\Lambda} \left(\textstyle\sum_m \mathbb{E}\left[w_m\right] \boldsymbol{\Psi}_m^\top \right) \mathbf{t}, \\ \boldsymbol{\Lambda} &= \left(\mathbf{B} + \sigma^{-2} \textstyle\sum_{m_1} \sum_{m_2} \mathbb{E}\left[w_{m_1} w_{m_2}\right] \boldsymbol{\Psi}_{m_1}^\top \boldsymbol{\Psi}_{m_2} \right)^{-1}, \end{aligned} \tag{11}$$

where $\boldsymbol{\Psi}_m \in \mathbb{R}^{N \times K}$ is the mth matrix slice along the 2nd dimension of the kernel design tensor $\boldsymbol{\mathcal{K}}$.

Step 3: Optimize $\boldsymbol{\alpha}_*$

$$\alpha_m^* = \begin{cases} \frac{a_m^2}{b_m^2 - a_m} \text{if } b_m^2 > a_m, & a_m = \phi_m^\top \mathbf{C}_{-m}^{-1} \phi_m \\ \infty \quad \text{otherwise}, & b_m = \phi_m^\top \mathbf{C}_{-m}^{-1} \mathbf{t} \end{cases}, \tag{12}$$

where $\mathbf{C}_{-m} = \sigma^2\mathbf{I} + \sum_{i \neq m} \alpha_i^{-1} \phi_i \phi_i^\top$ and $\phi_m \in \mathbb{R}^N$ with $\phi_m(n) = \sum_k v_k \kappa_k(\mathbf{x}_n, \mathbf{x}_m)$.

Step 4: Optimize $\boldsymbol{\beta}_*$

$$\beta_k^* = \begin{cases} \frac{c_k^2}{d_k^2 - c_k} \text{if } d_k^2 > c_k, & c_k = \psi_k^\top \mathbf{D}_{-k}^{-1} \psi_k \\ \infty \quad \text{otherwise}, & d_k = \psi_k^\top \mathbf{D}_{-k}^{-1} \mathbf{t} \end{cases}, \tag{13}$$

where $\mathbf{D}_{-k} = \sigma^2\mathbf{I} + \sum_{i \neq k} \beta_i^{-1} \psi_i \psi_i^\top$ and $\psi_k \in \mathbb{R}^N$ with $\psi_k(n) = \sum_m w_m \kappa_k(\mathbf{x}_n, \mathbf{x}_m)$.

Step 5: Optimize σ_*^2

$$\sigma_*^2 = \tfrac{1}{N} \mathbb{E}_{\mathbf{w}, \mathbf{v}} \left[\|\mathbf{t} - y(\mathbf{X}; \mathbf{w}, \mathbf{v})\|^2 \right]. \tag{14}$$

Summary: Our DSRVM algorithm is summarized in Alg. 1. Interleaving the updates of $q(\mathbf{w})$ and $q(\mathbf{v})$ will improve the approximation in (7). We first update $\boldsymbol{\alpha}$, followed by r updates of $q(\mathbf{w})$ and $q(\mathbf{v})$. Then we update $\boldsymbol{\beta}$, followed by r updates of $q(\mathbf{v})$ and $q(\mathbf{w})$. Each of the above $q(\mathbf{w})$ and $q(\mathbf{v})$ updates is followed by a σ^2 update. Any other order of the updates would be valid, however this order has been chosen for several reasons: (1) Part of the statistics that is neccessary for updating σ^2 is already calculated at the $q(\mathbf{w})$ and $q(\mathbf{v})$ steps, therefore we can follow with a σ^2 update at low cost. (2) The $\boldsymbol{\alpha}$ step depends only on $(q(\mathbf{v}), \sigma^2)$ and not on $(q(\mathbf{w}), \boldsymbol{\beta})$. Therefore any $(q(\mathbf{w}), \boldsymbol{\beta})$ update immediately before the $\boldsymbol{\alpha}$ step would be inefficient. The same reasoning holds for any $(q(\mathbf{v}), \boldsymbol{\alpha})$ update immediately before the $\boldsymbol{\beta}$ step. (3) Interleaving r updates of $q(\mathbf{w})$ and $q(\mathbf{v})$ between the $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ updates improves robustness, because it improves the approximation in (7) and hence the approximation of $\delta(\mathbf{w})$ and $\delta(\mathbf{v})$.

Initialization: First we initialize σ_*^2 with the variance of the targets $\mathbf{t}$. Then we select a single basis and a single kernel, i.e. setting all α_m and β_k to infinite except one. The selection process first calculates the inner product between $\mathbf{t}$ and all possible single basis/kernel combinations. Then we select randomly from the 50% of (m, k) pairs with the largest inner product. The optimal α_m for the selected (m, k) pair can be calculated in closed form when assuming $v_k = 1$ and β_k can be calculated for $w_m = 1$.

Complexity: The space and time complexity of the DSRVM algorithm depends highly on the number of relevance vectors $M_{\text{rel}} = |\{\alpha_m : \alpha_m < \infty\}|$ and the number of relevance kernels $K_{\text{rel}} = |\{\beta_k : \beta_k < \infty\}|$. Due to the sparsity constraints $M_{\text{rel}} \ll \max(M, K)$ and $K_{\text{rel}} \ll \max(M, K)$. Then the time complexity of all five training steps is in $O(M_{\text{rel}}^3 + K_{\text{rel}}^3 + M_{\text{rel}}^2 K_{\text{rel}}^2 N + M^2 N + K^2 N)$. The space complexity is in $O(MKN + M_{\text{rel}}^2 K_{\text{rel}}^2)$, i.e. it is mainly influenced by the $M \times K \times N$ gram matrix. Testing only involves the evaluation of (2) once, i.e. the time and space complexity is both in $O(M_{\text{rel}} K_{\text{rel}})$.

Algorithm 1: DSRVM learning algorithm

```
initialize q(w), q(v), α, β, σ²
while not converged do
    update α as in Step 3
    update q(w) as in Step 1 and σ² as in Step 5
    for r times do
        update q(v) as in Step 2 and σ² as in Step 5
        update q(w) as in Step 1 and σ² as in Step 5
    end for
    update β as in Step 4
    update q(v) as in Step 2 and σ² as in Step 5
    for r times do
        update q(w) as in Step 1 and σ² as in Step 5
        update q(v) as in Step 2 and σ² as in Step 5
    end for
end while
return q(w), q(v)
```

Next, we derive the predictive distribution for new data. Therefore, we need to solve (8), which is possible because it is a convolution of Gaussians:

$$p(t_{\text{new}}|\mathbf{t}, \mathbf{x}_{\text{new}}) \approx \mathcal{N}(t_{\text{new}}; y_{\text{new}}, \sigma^2_{\text{new}}), \tag{15}$$

where $y_{\text{new}} = y(\mathbf{x}_{\text{new}}; \mathbb{E}[\mathbf{w}], \mathbb{E}[\mathbf{v}])$. We need only y_{new} to make predictions and do not compute σ^2_{new}.

4 DSRVM vs. RVM

Our DSRVM extends RVM [47], [48]. Standard kernel-based methods, including RVM, pre-define the kernel function before learning and thus cannot identify and account for relevant facial parts. Consequently, RVM is bound to confuse distinct facial expressions sharing the same movements of specific facial parts.

The key difference between our DSRVM and RVM is that RVM uses a single, unique kernel for regression, centered at each training data point:

$$y_{\text{RVM}}(\mathbf{x}; \mathbf{w}) = \textstyle\sum_{m=1}^{M} w_m \kappa(\mathbf{x}, \mathbf{x}_m). \tag{16}$$

RVM seeks to learn a small subset of non-zero weights w_n associated with relevance vectors $\mathbf{x}_n$. By comparing (2) and (16), it follows that RVM does not have an explicit mechanism for additionally enforcing sparsity over the features of relevance vectors.

The RVM assumes a Gaussian distributed noise and independently distributed targets as in (3), while y is defined as in (16). The prior of $\mathbf{w}$ is defined as in (4). The RVM kernel is fixed and hence there are no kernel weights $\mathbf{v}$ included in the model. As a result, learning of RVM simplifies only to maximizing the marginal likelihood $\mathcal{L}_{\text{RVM}} = p(\boldsymbol{\alpha}_*, \sigma^2_*|\mathbf{t})$, under the assumptions that the prior of $(\boldsymbol{\alpha}_*, \sigma^2_*)$ is uniform. To this end, learning of RVM iterates three steps until convergence of $\mathcal{L}_{\text{RVM}}$:

Step a: Re-estimate $p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}_*, \sigma^2_*)$,
Step b: Optimize $\boldsymbol{\alpha}_*$,
Step c: Optimize σ^2_*.

For a detailed explanation of the RVM update steps and a comparison with DSRVM, see Appendix C.

When comparing both sets of update formulas, we see that the RVM update steps a, b and c correspond to the DSRVM update steps 1, 3 and 5, and indeed our DSRVM algorithm includes the RVM algorithm as a special case for a single kernel, i.e. for $K = 1$.

The RVM predictive distribution is a convolution of two Gaussians and thus can be computed in closed form:

$$p(t_{\text{new}}|\mathbf{t}, \mathbf{x}_{\text{new}}) = \mathcal{N}(t_{\text{new}}; y_{\text{new}}, \sigma^2_{\text{new}}), \tag{17}$$

with $y_{\text{new}} = y_{\text{RVM}}(\mathbf{x}_{\text{new}}; \mathbb{E}[\mathbf{w}])$. From comparing (16) and (2) we see that the RVM predictive function y_{RVM} is linear in $\mathbf{w}$, while the DSRVM predictive function y is multi-linear in $\mathbf{w}$ and $\mathbf{v}$.

5 DSRVM vs. Related MKL Methods

Our DSRVM is related to methods for Multiple Kernel Learning (MKL), where the goal is to learn an optimal way to combine kernel functions [11]. Existing MKL methods are mainly aimed at classification problems. Only a few MKL methods address regression [3], [34], [35], [46], [49]. To this end, the method of [34] uses a domain-specific heuristic, which is not generalizable to other domains, and thus seems unsuitable for our purposes. The methods of [35], [46] jointly learn SVR and kernel weights via semi-infinite linear programming [46], and gradient descent [35], and thus induce prohibitively long running times. By contrast, our DSRVM uses a computationally efficient EM algorithm, and significantly reduces running time of learning relative to the existing regression MKL methods. In addition, the above related work enforces sparsity only in the primal domain, without regularizing the total number of resulting relevance vectors. Our DSRVM is doubly sparse by identifying only a few relevant kernels and a few relevance vectors. The additional sparsity in the kernel domain leads to (1) improved runtime, since fewer kernels need to be evaluated and (2) improved generalization ability, since potentially uninformative kernels can be pruned out.

SMKL [35] defines the regression function as in (2), with the additional constraint of convex kernel combinations, i.e. $\sum_m v_m = 1$ and $v_m \geq 0$. In contrast to DSRVM, the SMKL method does not optimize the basis weights $\mathbf{w}$ and $\mathbf{v}$ within a Bayesian setting, but rather solves a max-margin formulation equivalent to a SVM. The SVM algorithm provides an optimal solution for the basis weights $\mathbf{w}$ given a fixed kernel and the kernel weights $\mathbf{v}$ are optimized by steepest descend. Unfortunately, the evaluation of the descend direction involves repeated executions of the SVM algorithm. Therefore SMKL repeats the SVM algorithm within a nested loop, leading to a large number of repetitions and thus a long training time. Furthermore, the sparsity of the kernel weights $\mathbf{v}$ is only encouraged by the convexity constraint. This is a weaker constraint than the hierarchical prior of the DSRVM, since it only limits the sum of all weights, and thus does not enforce specific weights to be zero. The SMKL training step includes a gradient evaluation with $O(KM^2_{\text{Rel}})$ and SVM solving with $O(M^3_{\text{Rel}} + M^2N)$. As for DSRVM, the complexity highly depends on the number of support vectors M_{Rel} and if $M_{\text{rel}} \ll M$, then the dominating term is $O(M^2N)$, which is similar to DSRVM, see Sec. 3. In practice, M_{Rel} for DSRVM is lower than for SMKL, and thus the DSRVM training is faster, see the results in Tab. 3.

The multi-class and multi-kernel RVM (mRVM) [6] is a RVM extension for classification that defines a shared hierarchical prior $\boldsymbol{\alpha}$ over the basis weights $\mathbf{w}$ for each class. Additionally, mRVM learns the kernel weights $\mathbf{v}$ for a convex combination of kernels. mRVM uses similar update formulas as RVM for $\mathbf{w}$ and additionally optimizes $\mathbf{v}$ by a Quadratic Programming algorithm. As in the case of SMKL, the sparseness of $\mathbf{v}$ is only weakly enforced by the convexity constraint, in contrast to the Bayesian

formulation of our DSRVM.

The multi-kernel RVM approaches of [3] and [49] use the same regression MKL formulation as ours (2), but combine the basis and kernel weights so that there is a separate weight for each basis and kernel combination. This leads to a large number of weights to learn (MK in comparison to our $M + K$), which makes the method more prone to overfitting and slower to train. We compare our method to this kernel formulation, see Sec. 7. While [3] uses the standard RVM to learn the weights, [49] formulates an efficient computation in the Fourier domain for circulant gram matrices. However, this is only possible because their particular application domain is significantly different from ours, since they seek to predict pixel values from a single image. In the application of this paper however, the features of each training instance stem from different images and thus the resulting gram matrices are not circulant.

6 Feature Extraction

This section motivates and describes the facial features that we use for facial behavior estimation using DSRVM.

Each of the used datasets provides annotated facial points. Details about the annotation process are explained in the corresponding database description. Given the points, we first align and normalize faces in each video frame to a canonical view. This view is obtained by a piece-wise affine warp to a base shape using the standard active appearance model (AAM), see [24] for details. The base shape has a size of 128×118 pixels for ShoulderPain and DISFA, and 128×155 pixels for SEMAINE. The AAM used for tracking SEMAINE had a different aspect ratio and we scaled the base shape to match vertically. A concatenation of all frames within a video sequence of length L results in a space-time volume of the size 128×118×L (or 128×155×L). We divide the space-time volume into subvolumes, and extract video features from each subvolume. In this way, we enforce that our video features are local, extracted from relatively small spatiotemporal supports, rather than from the entire face. As mentioned in Sec. 1, our local extraction of video features is motivated by a number of psychological studies [31], [39] which argue that facial expressions are characterized by distinct combinations of local FAUs, rather than global features extracted from the entire face. Since the right space-time location and scale of subvolumes that are relevant for facial behavior estimation are not known a priori, we extract the video subvolumes from a range of spatial and temporal scales. Specifically, we partition each video frame into a regular grid of S×S patches. In our experiments, we use S$\in \{6, 9\}$, i.e. the face is divided into 36 or 81 patches. Note that each of these patches defines a video subvolume with L frames. For analyzing various temporal scales, we scan these subvolumes along the time axis. The scan has a step size of one frame and a window size of T$\in \{1, 10, 20\}$ frames. Thus, we extract features from a total of S×S×(L-T+1) subvolumes per video and each feature vector includes information from a window of T consecutive frames.

As features, we use Local Binary Patterns (LBP) [30]. LBP is a histogram of local image intensity variations within a small pixel neighborhood. The features are defined for image patches (time-scale $T = 1$). For video subvolumes with $T > 1$, we use the temporal extension: LBP in three orthogonal space-time planes [55]. LBPs and their temporal extensions have been demonstrated useful for facial expression recognition [18]. Temporal extensions of LBP typically improve performance in comparison to the static LBP [18].

7 DSRVM Kernels

The previous section defines feature vectors $\{\mathbf{x}_k : k = 1, \ldots, K\}$ locally extracted from $K = S^2$ video subvolumes per frame window. In this section, we specify how to kernelize these features for DSRVM regression.

From (2), given a window with features $\mathbf{x}$ and mth training window with features $\mathbf{x}_m$, DSRVM uses K RBF kernels defined for each of their respective subvolumes $k \in \{1, ..., K\}$:

$$\kappa_k(\mathbf{x}, \mathbf{x}_m; \sigma_k) = \exp\left(-\frac{\|\mathbf{x}_k - \mathbf{x}_{mk}\|^2}{2\sigma_k^2}\right), \qquad (18)$$

where $\mathbf{x}_k$ and $\mathbf{x}_{mk}$ are the local features extracted from kth subvolume of the two frame windows.

Each kernel parameter σ_k^2 is estimated independently on training videos over the corresponding features $\mathbf{x}_k$. For fair comparison of DSRVM with alternative algorithms, we use the same set of kernels.

It is possible to use each of the K kernels as a separate basis function for RVM like in [3], [49], which results in MK basis functions and thus in a kernel gram matrix of size $N \times (MK)$. We compare with this approach and call it RVM separate (RVM sep). Note that RVM sep uses the standard RVM algorithm as in [3], since the gram matrix is not circulant and thus the more efficient method of [49] cannot be applied.

8 Results

We evaluate our DSRVM on four datasets: (1) the artificial dataset used for benchmark evaluation of regressors [46, p. 1548]; (2) the UNBC-MacMaster Shoulder Pain Expression Archive Database (ShoulderPain) [24]; (3) the Denver Intensity of Spontaneous Facial Action (DISFA) database [25]; and (4) the SEMAINE database [26]. We chose (1) because it has been used by competing MKL regression methods, (2) because it is the only dataset that provides pain *intensity*, and (3) and (4) because they contain non-posed and time-continuously annotated videos.

For each database, we measure the performance of DSRVM and competing methods and provide further statistics: the selected number of relevance vectors (#RV), relevant kernels (#RK), training time, and testing time. Additionally, we visualize the selected kernels for an intuitive interpretation of the learned model.

8.1 The Artificial Dataset

Here we use the Sonnenburg et al. [46] (page 1548) regression experiment, which is designed to evaluate the kernel adaption ability of an algorithm. The task is to learn the target function $t = \sin(fx) + \sin(x) + x + \epsilon$, where f is the frequency of a varying Sine function, $f \in \{1, 2, \ldots, 20\}$, and ϵ is white Gaussian noise with variance 0.3. The set of kernels consists of 10 RBF with the length-scale parameters $\gamma_k \in \{0.001, 0.005, 0.01, 0.05, 0.1, 1, 10, 50, 100, 1000\}$. The range of length-scale values for the RBF is chosen to correspond to different frequencies of the Sine function, so that an optimal MKL algorithm needs to adapt the used Kernel to the current frequency. Each feature-target pair (x, t) with $x \in \mathbb{R}$ is constructed by randomly sampling x from a uniform distribution in $[0, 10]$. We use 2000 feature-target pairs (x, t), where one half serves for training, and the other half for testing.

8.2 The ShoulderPain Dataset

The ShoulderPain data [24] consists of videos showing faces of patients suffering from shoulder pain while moving their arms and thus performing range-of-motion tests. Facial expressions during two distinct arm movements are recorded: (1) the subject moves the arm her/himself, and (2) the subject's arm is moved by a physiotherapist. Only one of the arms is affected by pain, but movements of the other arm are recorded too as a control set. The videos show 200 sequences of 25 subjects, with a total of 48398 frames. 66 tracked facial landmarks are provided with the dataset. For each frame, discrete pain intensities are provided according to Prkachin and Solomon method [33]. The pain intensity is quantified into 16 discrete levels (0 to 15), annotated by the database creators. For the distribution of pain intensity levels see [24]. Since the vast majority of frames 40029 contains no pain (level 0), in training we remove most frames with pain level 0 from the beginning and the end of each video sequence, so that the remaining non-pain frames matches the number of frames with pain at level 2.

The ShoulderPain videos have been shown highly challenging in previous work on pain vs. no pain detection [23], [37], 4/5-level discrete pain classification [14], [38], and continuous pain estimation [19].

8.3 The DISFA Dataset

The DISFA dataset [25] contains spontaneous facial expressions of young adults while watching youtube videos. These videos are 9 short clips expected to elicit happiness, surprise, fear, disgust and sadness. 27 subjects were recorded with a resolution of 1024x768 pixels, and a frame-rate of 20 frames-per-second, resulting in a total number of 130754 frames. Each of these frames has been annotated with FAU's and their corresponding intensity on a 0-5 discrete scale by an expert FACS rater. The following FAU's are annotated: 1, 2, 4, 5, 6, 9, 12, 15, 17, 20, 25, and 26. For the intensity distribution of each FAU see [25]. 66 active-appearance-model (AAM) tracked facial landmark points are also provided for all data.

8.4 The SEMAINE Dataset

The SEMAINE dataset [26] contains spontaneous facial expressions of users having a conversation with an operator. The operator talks about a topic that is relevant to the user, and tries to elicit different emotions. The face of the user has been recorded with a resolution of 780x580 pixels, and a frame-rate of 50 frames per second. We use a subset of the SEMAINE dataset that is part of AVEC2012 [43] and contains 43 video sequences of 10 subjects, and a total number of 582235 frames. The SEMAINE session numbers of the subset are specified in Appendix A. The dataset provides annotations for the intensity of several affect dimensions on a continuous scale between -1 and 1. We use the annotations of valence and arousal, since they are relevant for discrimination between many affective states [40]. Each video is annotated per frame by six raters. The mean of the six raters is used as ground truth, leading to valence and arousal intensity distributions that are close to a Gaussian distribution with mean 0 and variance 1. This is a standard approach followed in other works [43]. An alternative would be to align all annotations and, in turn, handle delays and biases introduced by various annotators [28]. We opt for the former in order to make our results comparable to those of the related works [43]. The face has been tracked by the AAM model described in [7] and 113 facial landmarks are known for each frame.

8.5 Evaluation Metrics and Settings

Regression accuracy is evaluated using the mean squared error (MSE), and Pearson correlation coefficient (CORR), which are common within the field of human behavior estimation [12]. MSE and CORR are computed between our predicted target intensities and the ground truth. While MSE is better suited for measuring identity between signals, CORR is better suited for comparison across the datasets with different ranges due to its implicit scaling of the targets. Both metrics are equivalent when the targets and predictions are normalized to zero mean and variance 1. The targets for ShoulderPain (range 0-15) and DISFA (range 0-5) are discrete, but treated as continuous values without any modification during training and testing. E.g. for the discrete pain target value 1 and the prediction 1.367, the MSE would be $(1 - 1.367)^2$.

Recently, the Intra-Class Correlation Coefficient ICC(3,1) [44] has been proposed for evaluating approaches to human behaviour analsys (e.g. [14], [25]). Similar to CORR, ICC also provides an implicit scaling of the targets. To facilitate comparison to the related methods evaluated in terms of ICC, we test our performance using ICC as well, and present these results in Appendix D. We include MSE and CORR in the main paper, since they have been the most common metrics in affect analysis works [12], and MSE is the most common metric for evaluating regression algorithms (e.g. [35], [47]), but otherwise there is no preference over ICC.

For a pair-wise comparison of our DSRVM to other methods, we estimate significance of the results using a

two-tailed Student's t-test. We report the t-test probability value – p-value – that is minimally needed to reject the null hypothesis (i.e., DSRVM and the comparing method are the same), where low p-values correspond to high significance.

We also report the number of selected basis (# RV, for Relevance Vectors), the number of selected kernels (# RK, for Relevant Kernels), and the times for training (TRN) and testing (TST). TRN is the time needed for training the model on 2,000 data points and TST is the time needed for testing the model. Since the number of testing samples varies between folds and datasets, we divide TST by the number of samples per fold and multiply by 2,000 to represent the time for 2,000 data points. The running times are evaluated on a single core of an Intel Xeon E5-2640 CPU with 64 GB RAM. On average, 4GB of memory is needed for 2000 training examples. In order to get a robust result regarding local minima, we report the average of 10 random initializations as explained in the initialization paragraph of Sec. 3.

DSRVM training is iterative. When we evaluate performance w.r.t. MSE (or CORR), we use the MSE-based (or CORR-based) convergence criterion for stopping the iterations in training (see Alg. 1, line 2). This gives two variants of our DSRVM. In order to have a fair comparison with the other models, each of them is separately optimized regarding CORR and MSE.

For evaluation on the Artificial dataset, we use the standard setting of [46], [47]. Specifically, we randomly sample artificial data in order to form 10 sets of data. Each set (a.k.a, fold) is split in half for training and testing. The reported results are averaged across the 10 folds. For the non-artificial datasets ShoulderPain, DISFA and SEMAINE, we use the subject-independent setting, where the videos of selected subjects are left out for testing, and the videos of all other subjects in the dataset are used for training. This process is repeated with different subjects, until all subjects have been used for testing. The results are combined by calculating the weighted average across all subjects left out for testing. The weight of each subject corresponds to the number of frames each subject occurs in. We use all frames of testing videos and equidistantly sub-sample 2000 frames for training to reduce the SMKL training time below 5 hours. For the AU recognition experiments, we additionally assure that at least 25% of the training data contains the specific AU which we train for. The following paragraph explains how we divide the available data into training and testing videos.

For the ShoulderPain dataset we do a full evaluation of all space-time scales S=$\{6, 9\}$ and T=$\{1, 10, 20\}$. Since the differences between scales was rather low, we use a simple baseline and do not account for the temporal extent of changes in facial features (i.e. S=6 and T=1) for the DISFA and SEMAINE datasets.

8.6 Baseline Methods

We compare our DSRVM with three baselines — namely, RVM [47], SMKL [35] and mRVM [6].

RVM is specified in Sec. 4. RVM uses a single kernel, and thus we cannot use the expression for DSRVM kernels given by (18). Hence, we use three strategies to compute the RVM kernel. The first strategy, called *RVM-all*, computes the kernel as a sum of *all* DSRVM kernels given by (18) with kernel weights $v = 1$. The second strategy, called *RVM-best*, sets the kernel as one of the DSRVM kernels given by (18) that gives the best CORR result — it sets the corresponding weight in v to 1 and all others to 0. The third strategy, called *RVM-sep*, sets one dimension in v to 1 and all others to 0 for all possible K dimensions. This leads to MK basis functions, in contrast to M basis functions of the other approaches.

SMKL is well suited for our comparison, since its inference model is the same as that for DSRVM: given the kernel gram-matrix, the estimated target is calculated in a multi-linear operation weighted by the basis and the kernel weights. Furthermore, SMKL is based on support vector regression (SVR) [45], the main competing regression method for RVM [47]. The SVR regression parameter ϵ and cost C have been optimized by a grid-search on training data. For implementing RVM we use the SparseBayes Matlab toolbox [47], and for implementing SMKL (i.e., SVR) we use the LIBSVM [2].

mRVM is a multi-class multi-kernel classifier and thus this experiment compares a classifier with continuous regression models. Specifically, we compare with the mRVM-1 as defined by [6], since it is rather similar to DSRVM due to the constructive approach that starts from a single basis function. The targets of the SEMAINE and artificial data are continuous, and thus it is necessary to convert them into classes for running mRVM. We discretize the targets into c classes by dividing the range into c equidistant bins. An inverse transform from the predicted class to a continuous value is needed for evaluation and thus we map each predicted class to the center value of the corresponding bin. The mRVM performance is evaluated using the optimal c yielding the best CORR and MSE results of mRVM. Additionally the results for varying c are provided in Appendix E. In contrast, the targets of the ShoulderPain and DISFA datasets are discrete, and thus no further discretization is needed for testing mRVM.

8.7 Results on the Artificial Dataset

We conduct the Sonnenburg et al. [46, p. 1548] regression experiment for comparing the kernel choices made by DSRVM and those made by SMKL. For the target artificial dataset, both DSRVM and SMKL use 10 RBF kernels, whose widths γ_k are specified in Sec. 8.1. The results are shown in Fig. 2. As the frequency of the target function changes, DSRVM adapts the kernel weights so as to tune to the particular frequency. As can be seen, DSRVM learns both positive and negative kernel weights. For DSRVM, negative weights (blue) are usually paired with positive weighted kernels (red) of similar width γ. Starting with the frequency 1, DSRVM chooses kernel widths 1 and 100. As the frequency increases, the kernel width is shifted toward

lower values until the main width is 0.01 for the frequency 20. In contrast, kernel weights learned by SMKL are only positive. From Fig. 2, SMKL always selects the smallest width of 0.001, which leads to higher number of RV's and higher risk of overfitting (see also Fig. 3).

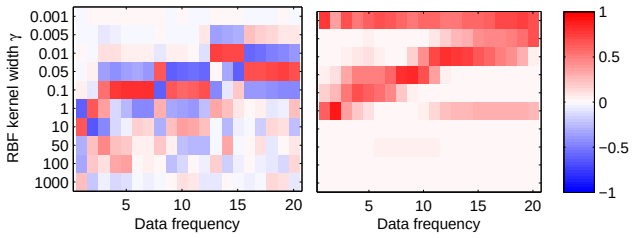

Fig. 2: Kernel weights learned by DSRVM (left) and SMKL (right) on the Sonnenburg artificial data [46] for the varying frequency of the target function.

Fig. 3 (left column) compares DSRVM with SMKL, RVM-all and RVM-sep, in terms of their MSE and CORR rates, as well as the number of selected relevance vectors (# RV) and relevant kernels (# RK) on the Artificial dataset. As can be seen, DSRVM yields better MSE and CORR rates, and selects significantly fewer RV's than SMKL and RVM-all. The MSE depends on the scale of the targets. Since the artificial data has a smaller scale than ShoulderPain, it also has lower MSE results. Note that the range of the MSE results for the artificial data at frequency 20 is from 0 to 0.06, and thus the DSRVM improvement of 0.02 accounts for 33% of the range, which makes the improvement significant.

SMKL selects fewer RKs, but its performance is worse than that of DSRVM, since it selects the smallest kernel width (see Fig. 2). Selecting smaller kernel widths allows for more fine grained modeling, but then more RVs are needed. An optimal algorithm selects the width just small enough to model the target function. If unnecessary small widths are selected, then too many RVs are needed and the algorithm is prone to overfitting. DSRVM selects in this case more RKs than SMKL, but the kernel widths are better adjusted to the data, as is obvious from the results.

8.8 Results on the ShoulderPain Dataset

We carry out two sets of experiments on the ShoulderPain data aimed at testing how (i) the number of training examples, and (ii) changes in space-time scale, affect the performance of the tested models.

Fig. 3 (right column) compares results to those of SMKL, RVM-all and RVM-sep for a varying number of training examples. The space-time scale of extracting video features is fixed at a regular grid of 6×6 patches (S=6) per frame, and temporal window of 1 frame (T=1). As can be seen, our accuracy is better in terms of MSE than it is the case for the competing approaches, and the CORR results for DSRVM are on par with those for SMKL. The sparse kernel prior of DSRVM brings less advantage in this latter case, since the facial expression of pain involves both, the upper and lower face and thus is less localized than e.g. specific FAUs.

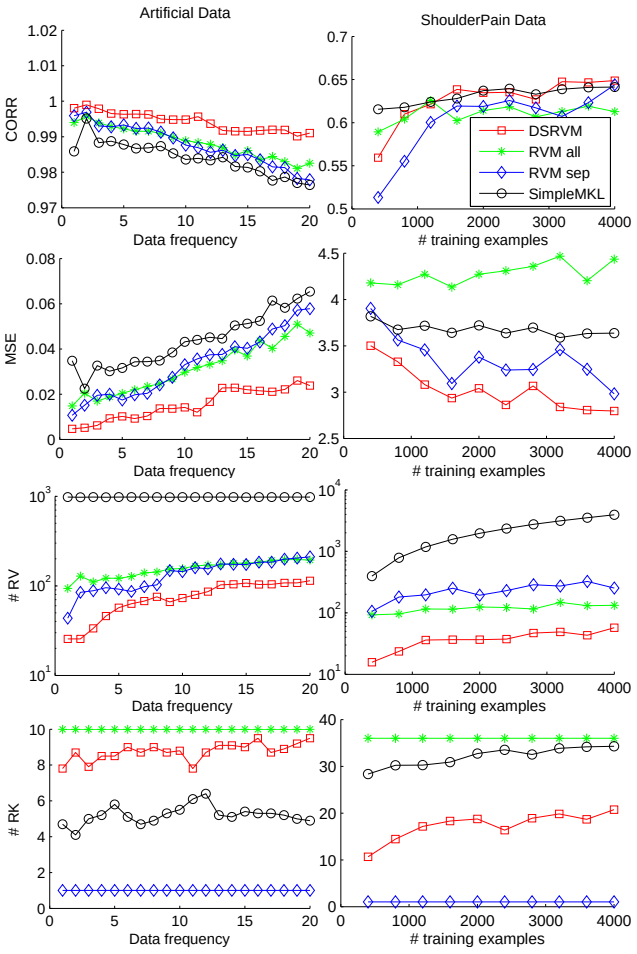

Fig. 3: Results on the artificial data (left column) for a varying frequency of the target function and results on the ShoulderPain data (right column) for the pain targets and a varying number of training examples: CORR (1st row), MSE (2nd row), the number of selected relevance vectors (#RV) (3rd row), and the number of selected relevant kernels (#RK) with non-zero kernel weights (4th row). Note that #RV is shown on the logarithmic scale.

Fig. 3 also shows the number of relevance vectors (RV), and the number of relevant kernels (RK) with non-zero kernel weights learned by DSRVM, SMKL, RVM-all and RVM-sep, as the number of training examples increases. Note that RVM cannot select kernels, therefore the graphs for RVM-all and RVM-sep stay constant at the total number of kernels. As can be seen, DSRVM consistently selects fewer RVs and RKs than the other methods. This suggests that the doubly sparse formulation of DSRVM achieves greater sparsity of kernels than the compared methods. In addition, since DSRVM selects significantly fewer RV's than SMKL, DSRVM regression is more computationally efficient than that of SMKL.

Tab. 1 (left column) compares the results by DSRVM to those by RVM, SMKL and mRVM for different space-time scales. The number of training examples is fixed at 2000. In terms of CORR, DSRVM outperforms SMKL and all RVM variants for the spatial scale set to 9×9 patches

TABLE 1: Results on the ShoulderPain data for the pain targets (left column) and on the SEMAINE data for the arousal (Ar.) and valence (Val.) targets (right column). For ShoulderPain, features use different space-time (S-T) scales, and for SEMAINE S=6 and T=1. p-value measures significance in comparison to DSRVM. The table shows mean squared error (MSE) and the correlation with the targets (CORR). The best results are marked bold. If two results differ by at most 0.01 and the p-value is greater than 0.05, we mark both results bold. Note that the target intensity range for ShoulderPain (0 to 15) is larger than for SEMAINE (-1 to 1) and thus the MSE scores are not comparable across the datasets.

		ShoulderPain						SEMAINE	
	Method	S6x6 T1	S6x6 T10	S6x6 T20	S9x9 T1	S9x9 T10	S9x9 T20	Ar.	Val.
CORR	DSRVM	**0.63**	**0.66**	**0.63**	**0.63**	**0.63**	**0.62**	**0.31**	**0.31**
	RVM all [47]	0.61	0.62	0.61	0.60	0.59	0.58	0.25	0.23
	p-value	.22	.07	.18	.03	.08	.10	.02	.02
	RVM best [47]	0.29	0.45	0.45	0.39	0.49	0.46	0.20	0.18
	p-value	.00	.00	.00	.00	.00	.00	.03	.06
	RVM sep [3], [49]	0.62	0.62	0.60	0.62	0.56	0.59	0.21	0.23
	p-value	.16	.10	.11	.37	.02	.13	.01	.00
	SMKL [35]	**0.64**	**0.66**	**0.63**	**0.62**	0.60	0.59	0.22	0.22
	p-value	.49	.20	.14	.23	.07	.04	.01	.00
	mRVM [6]	0.53	0.52	0.45	0.49	0.50	0.50	0.10	0.12
	p-value	.00	.00	.00	.05	.02	.04	.00	.00
MSE	DSRVM	**3.04**	**2.86**	**3.22**	**3.00**	**3.15**	**3.08**	**0.042**	**0.058**
	RVM all	4.27	4.37	4.55	3.72	4.00	3.98	**0.046**	0.065
	p-value	.02	.01	.01	.03	.02	.01	.15	.01
	RVM best	9.85	4.90	3.79	4.16	4.32	4.84	0.057	0.073
	p-value	.03	.00	.01	.00	.00	.00	.00	.00
	RVM sep	3.38	2.89	4.00	3.37	3.99	4.61	0.049	0.073
	p-value	.64	.50	.30	.10	.14	.03	.03	.00
	SMKL	3.72	3.69	3.85	3.67	3.78	3.93	0.051	0.070
	p-value	.03	.02	.03	.03	.01	.01	.00	.00
	mRVM	4.30	4.46	4.49	4.41	4.49	4.58	0.059	0.072
	p-value	.01	.01	.01	.01	.01	.01	.00	.02

(S=9), and all temporal scales T=$\{1, 10, 20\}$. When the spatial scale is set to 6×6 patches (S=6), DSRVM yields a comparable performance to that of SMKL while the RVM variants perform worse. RVM-best is the worst performing with high significance (low p-values). This demonstrates that a single kernel is not sufficient for regression on the ShoulderPain dataset, i.e. a specific face patch is not sufficient to recognize the pain level. In terms of MSE, DSRVM outperforms all methods.

Tab. 1 also shows that all regression methods (except the single kernel RVM-best) perform better than the classification mRVM. Classification methods are disadvantaged when applied to intensity estimation, since the inherent value of intensities and their "greater than" and "equal" relationships are not incorporated in a classification model and thus each intensity is modeled as a class on its own.

From Tab. 1 (left column), for all methods, we observe that the temporal scales of T=1 and T=10 video frames give better results than T=20. This can be explained by research findings in psychology, which suggest that the intensity of pain experience can be encoded from the number of facial actions recruited and their vigor – lower levels of pain are manifested in brow lowering and narrowing of the eyes, while higher levels of pain are manifested by these actions expressed more vigorously and recruiting additional (lower face) actions [5]. Given that ShoulderPain videos have been recorded at 25 fps, and that facial muscle activation is relatively rapid (onset ranging from $1/16$ seconds to $1/3$ seconds [36]), a temporal window of 9–10 frames covers the onset of even the slowest facial change. Hence, longer temporal windows (say T=20) cover not only the current pain level but the subsequent one(s) too. Hence, using longer temporal windows leads to more frequent confusion between successive pain levels, as temporally consistent features are learned covering multiple pain levels rather than a single one (as clearly observable from Fig. 4 too).

Overall, the spatial scale of 6×6 patches gives the best results. This is, because a patch at the coarser spatial scale of 6×6 patches may be more robust to alignment errors or out-of-plane rotations. For all space-time scales, DSRVM pain level estimation results are better than those presented in our earlier work [19], where we reported CORR of 0.59.

Fig. 4 shows a sample video frame from the ShoulderPain dataset, and kernel weights v learned for different space-time scales by DSRVM and SMKL. Each patch of the video frame corresponds to one kernel. We observe that both DSRVM and SMKL select similar patches with large kernel weights as relevant for shoulder-pain-level estimation. These patches fall mainly on the facial areas around the eyes, nose and mouth corners. As already explained above, these results agree with the well-known definition of the facial expression of pain [52], Including brow lowering (FAU4) and narrowing of the eyes (FAU7) as well as additional facial action such as upward lip pull (FAU12). Fig. 4 also shows that DSRVM learns sparser kernel weights than SMKL (i.e., fewer patches are selected as relevant for pain-level estimation).

8.9 Results on the DISFA Dataset

Tab. 2 shows the results attained by DSRVM, SMKL, RVM-all, RVM-best and RVM-sep on the DISFA dataset, for different FAUs. Additionally, we compare to the method of [25] on our feature set. [25] learns a low-dimensional manifold with Spectral Regression (SR), followed by SVM classification. The SR step includes training and testing subjects and thus is not subject-independent. In order to have a fair comparison, we use the same subject-independent setting as for the other methods. We run SR followed by SVM (SR+SVM) and the results are shown in Tab. 2. Additionally, we provide a comparison to DSRVM within the same setting of [25] (i.e. subject-dependent) in the supplementary material (Appendix D).

As shown in Tab. 2, DSRVM gives the best CORR for most FAUs. For FAUs 2, 12, 20 and 25 the DSRVM is on par with the best result (the p-value is large in all cases). DSRVM and mRVM give the best MSE, while DSRVM is better than mRVM on average. The best MSE scores are reached for DSRVM at FAUs 5 and 15, however the CORR for the same FAUs are relatively low with 0.17 and 0.32.

TABLE 2: Results on DISFA for FAU and S=6, T=1. AVG is the average results of all FAUs.

	Method	AU1	AU2	AU4	AU5	AU6	AU9	AU12	AU15	AU17	AU20	AU25	AU26	AVG
CORR	DSRVM	0.31	**0.28**	**0.54**	0.17	**0.57**	**0.43**	**0.80**	0.32	**0.40**	**0.23**	**0.66**	**0.42**	**0.43**
	RVM all	0.26	**0.29**	0.41	0.12	0.46	0.32	0.75	0.33	**0.40**	0.19	0.62	0.40	0.38
	p-value	.11	1.00	.00	.05	.02	.00	.05	.49	.80	.07	.34	.59	.01
	RVM best	0.18	0.15	0.46	**0.19**	0.40	0.30	0.73	0.18	0.21	0.16	0.61	0.17	0.31
	p-value	.00	.01	.07	.16	.03	.03	.01	.01	.00	.00	.11	.00	.00
	RVM sep	**0.35**	0.27	0.50	0.17	0.47	0.34	0.78	**0.34**	0.38	0.21	0.61	0.40	0.40
	p-value	.42	.24	.18	.68	.03	.03	.06	.16	.61	.14	.08	.49	.04
	SMKL	0.26	0.20	0.45	0.17	0.49	0.36	**0.81**	0.26	0.36	0.22	**0.66**	0.40	0.39
	p-value	.06	.01	.02	.94	.06	.02	.26	.05	.14	.43	.93	.47	.00
	mRVM	0.21	0.16	0.32	0.10	0.39	0.40	0.69	0.23	0.32	**0.24**	0.59	0.23	0.32
	p-value	.25	.17	.02	.22	.02	.57	.00	.22	.17	.97	.00	.03	.00
	SR+SVM	0.13	0.13	0.33	0.04	0.40	0.39	0.75	0.21	0.29	0.18	0.47	0.36	0.31
	p-value	.01	.00	.00	.02	.01	.24	.04	.08	.08	.14	.00	.09	.00
MSE	DSRVM	**0.66**	0.59	**0.84**	0.15	**0.42**	**0.42**	0.39	0.19	**0.31**	0.26	1.01	**0.45**	**0.47**
	RVM all	1.20	0.85	1.11	0.27	0.59	0.59	0.42	0.25	0.43	0.35	1.06	0.57	0.64
	p-value	.01	.02	.01	.03	.02	.00	.43	.01	.01	.03	.66	.03	.00
	RVM best	1.28	1.21	1.05	0.30	0.60	0.78	0.48	0.33	0.51	0.42	1.14	0.70	0.73
	p-value	.01	.01	.07	.11	.17	.08	.02	.07	.02	.09	.13	.00	.00
	RVM sep	1.10	0.85	1.09	0.40	0.64	0.67	0.42	0.26	0.45	0.43	1.15	0.73	0.68
	p-value	.01	.00	.10	.10	.00	.04	.06	.00	.01	.00	.24	.00	.00
	SMKL	0.73	0.69	0.92	0.17	0.49	0.46	**0.35**	0.23	0.34	0.24	**0.97**	0.50	0.51
	p-value	.22	.04	.07	.34	.03	.14	.06	.28	.14	.26	.55	.08	.03
	mRVM	**0.66**	**0.57**	1.35	**0.10**	0.57	**0.41**	0.56	**0.17**	0.33	**0.19**	1.36	0.62	0.58
	p-value	.99	.95	.03	.10	.04	.94	.00	.48	.36	.01	.00	.12	.07
	SR+SVM	1.00	1.13	1.52	0.20	0.69	0.50	0.59	0.52	0.45	0.44	1.66	0.58	0.77
	p-value	.13	.07	.01	.34	.01	.13	.00	.28	.02	.08	.00	.05	.00

TABLE 3: Model statistics on the DISFA data averaged over all FAU targets. Shown are the number of relevance vectors (#RV), the number of relevant kernels (#RK), training runtime (TRN) in sec$\times 10^2$, and test runtime (TST) in sec. #RK is only shown for models that adjust the kernel.

Method		#RV	#RK	TRN	TST
DSRVM		55.5	21.4	21.3	0.4
RVM all	[47]	120.6	-	1.2	1.3
RVM best	[47]	77.7	-	1.3	0.1
RVM sep	[3], [49]	217.6	-	36.2	3.1
SMKL	[35]	1914.2	32.5	149.8	7.8
mRVM	[6]	52.6	36.0	77.4	1.2
SR+SVM	[25]	464.2	-	0.1	0.1

TABLE 4: Comparison with previous work on the SEMAINE dataset. [43] only reports CORR and [29] only reports MSE.

	Method	Ar.	Val.
CORR	DSRVM	0.31	0.31
	Schuller et al. [43]	0.11	0.17
MSE	DSRVM	0.042	0.058
	Nicolaou et al. [29]	0.058	0.020

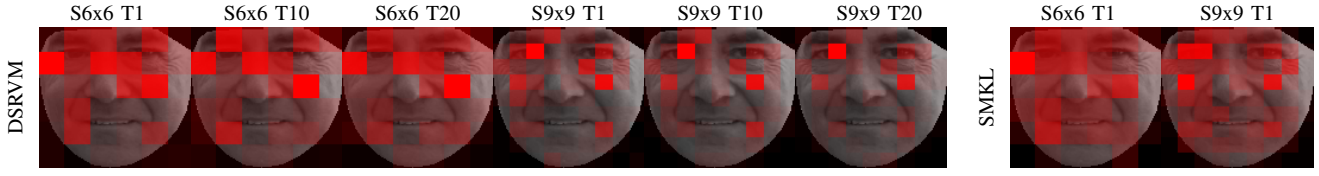

Fig. 4: ShoulderPain dataset: The values of kernel weights v learned by DSRVM (left) for various spatial (S) and temporal (T) scales are indicated by the intensity of color red of the corresponding patches. Each patch corresponds to one kernel, and the larger the kernel weight the redder the patch. The reddest patches correspond well with the FAU definitions presented in [8]. Additionally we show the learned kernel weights by SMKL (right) for selected scales.

This difference within CORR and MSE stems from the bias of the FAU intensity distribution within the DISFA data. FAUs 5 and 15 occur rarely within the data in comparison with e.g. FAU4. Therefore a model can reach a good MSE by conservatively rating closer to intensity 0, even if a few high intensity FAU events are missed. In contrast to that, CORR is a relative measure and highly penalizes the score if high intensity FAU events are missed. Therefore CORR and MSE show different trends, since they measure different aspects of the differences between predictions and targets. The same effect can be seen in Fig. 3, where DSRVM is on par with SimpleMKL regarding CORR but outperforms SimpleMKL regarding MSE. This can also explain the differences between DSRVM and mRVM. mRVM and DSRVM have almost the same performance in terms of MSE, while DSRVM is clearly better with respect to CORR. One reason as to why mRVM is unable to robustly learn higher-intensity levels is it treats each level as a separate class, while data samples for higher-intensity levels are scarce in our domain. In contrast, DSRVM fuses all levels together to a common regression function, and thus is able to compensate for missing or scarce data samples at certain levels.

Tab. 3 provides average statistics of the learned models for CORR, the statistics for other measures are similar (see Appendix F). The table shows the number of relevance vectors (#RV), the number of relevant kernels (#RK), training runtime (TRN) in sec$\times 10^2$, and test runtime (TST) in sec. #RK is only shown for models that adjust the kernel weights. Tab. 3 shows the advantages of DSRVM relative to SMKL and RVM, in terms of #RV and #RK. DSRVM and mRVM select significantly fewer RV's than RVM and SMKL. In terms of RK's, DSRVM uses fewer kernels than SMKL and mRVM. Specifically, it selects two thirds of the kernels selected by SMKL as being relevant for regression, thereby achieving twice greater sparsity than SMKL. Note that the sparse kernel and basis selection of DSRVM directly affects the test running time (TST). DSRVM regression is about 20 times faster than that by SMKL, and even faster than RVM-all. Moreover, DSRVM and mRVM also have 7 times faster training time (TRN) in comparison to SMKL. As expected, the training time of RVM-all and

RVM-best is lower than that for DSRVM, because these methods learns only the basis weights, whereas the kernel weights are fixed. SR+SVM training is fast since the SVM is applied to the low-dimensional manifold. However, the performance is relatively low, which is probably caused by overfitting the manifold to the training subjects. The SR+SVM results are much lower than in [25], due to the subject-independent evaluation.

Fig. 5 shows the kernel weights v learned by DSRVM and SMKL for different FAUs on DISFA data. The facial regions selected as relevant for FAU detection correspond well with the FAU definitions presented in [8]. For FAUs 17 and 20, we see that DSRVM is more sparse than SMKL, although the emphasize lies on similar regions.

In order to analyze the information content of different patches, we repeatedly apply DSRVM for varying number of patches. For each run, only the most relevant patches defined by their respective kernel weight v are used for training. The results for FAU12 are shown in Fig. 6. There is a sharp performance raise for the first patches, which reaches a peak between 3 to 12 patches. Then the performance slowly decreases for larger number of patches. This confirms previous work [22], [57], which found a sparse patch subset to be sufficient for recognizing FAUs and that the inclusion of further information leads to lower performance.

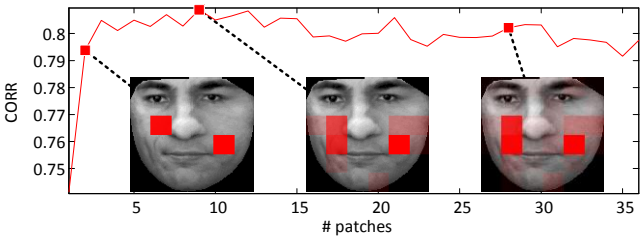

Fig. 6: DSRVM results measured by CORR for different numbers of facial patches. Shown is FAU12 from DISFA as example. Additionally, the learned kernel weights v are shown for 2, 9 and 28 patches.

8.10 Results on the SEMAINE Dataset

Tab. 1 (right column) shows results attained by DSRVM, SMKL, RVM-all, RVM-best and RVM-sep on the SEMAINE dataset for arousal and valence targets. DSRVM significantly outperforms the other methods for both valence and arousal. Similar to the results on the other datasets, DSRVM learns fewer RV's and RK's than SMKL.

Fig. 7 shows a sample video frame from the SEMAINE dataset, and kernel weights v learned by DSRVM and SimpleKLM, for a given value of arousal and valence. For arousal, DSRVM focuses more on the facial area around the nose and below the eyes. This can be explained by the fact that high arousal (such as in surprise, disgust and happiness) is characterized by vertical facial motions in those areas (e.g. nose wrinkling in disgust and raised cheeks in happiness). For valence, DSRVM focuses on the inner eyebrows, the nasolabial furrow and the eye corners. Again, this can be explained by the facial motion being typical for positive valence (happiness, characterized by smiles that affect the nasolabial furrow and the eye corners) and for negative valence (e.g. frowns and deepened nasolabial furrow like in anger). SimpleMLK is less sparse, and regards almost all patches on the entire face as relevant for regression, including the patches learned by DSRVM. The focus areas are different, which can be caused by the non-sparse weights.

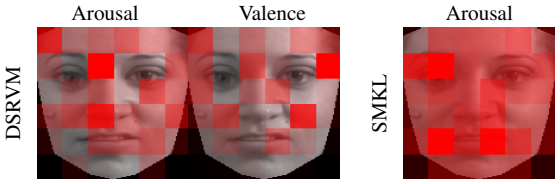

Fig. 7: A sample video frame from the SEMAINE dataset. See the caption of Fig. 4. Additionally, we show the SMKL results for arousal, which show that DSRVM learns sparser kernel weights.

Tab. 4 compares DSRVM with prior work [29], [43] on the SEMAINE datasets. Note that the comparison in Tab. 4 is not standard, since prior work uses different subsets of SEMAINE. But, since each subset is supposed to represent the entire dataset reasonably well, the results in Tab. 4 can be viewed as a reasonably good estimate of a standard comparison. In particular, [29] uses tracked facial points as features, and an output-associative RVM for regression. Results are reported only as root MSE and separately per subject for positive and negative arousal/valence sequences. To compare the results with ours, we raise them to the power 2 and take the average over subjects and positive/negative classes. [43] uses LBP histograms as features, a different face alignment from ours, and SVR for regression. The results are reported only as CORR and we take the average of the test and development set from the fully continuous sub-challenge. Tab. 4 shows that DSRVM outperforms [43] for both, valence and arousal. Regarding [29], DSRVM is better for arousal, but not for valence. However, [29] trains separate models for previously detected positive and negative classes and thus has an information advantage.

9 Conclusion

We have addressed estimation of continuous-valued intensities of facial expressions – a problem that has received scant attention in prior work – within the regression framework. Motivated by psychological studies on the importance of local features for facial behavior, we have specified a new regression method – called Doubly Sparse Relevance Vector Machine. DSRVM generalizes RVM by jointly choosing a sparse set of relevant kernels associated with face parts, and a sparse set of relevance vectors (i.e., training data) for modeling facial expressions. This also advances related multiple-kernel learning (MKL) methods, typically specified within the max-margin framework, where enforcing joint sparsity of kernel weights and relevance vectors is

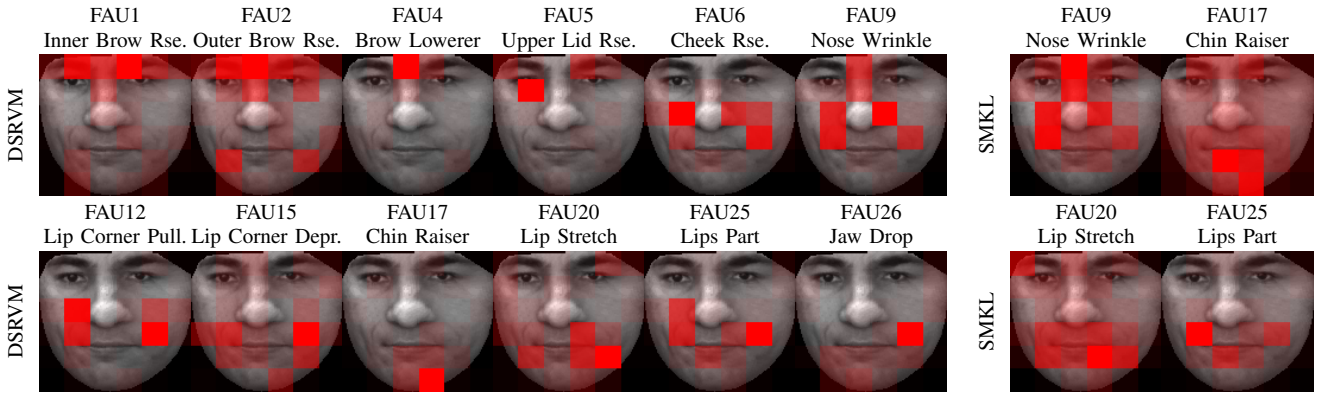

Fig. 5: DISFA data: The values of kernel weights v learned by DSRVM (left) and SMKL (right) for different FAUs. See the caption of Fig. 4. Note: the facial image serves for region identification, it does not show activation of target FAUs.

difficult. DSRVM uses efficient EM algorithm for learning relevant kernels and relevance vectors, and thus achieves about 20 times faster training than one of the latest MKL methods, called SMKL. Also, due to achieving higher sparsity, DSRVM has more than 3 times faster test runtimes, and more economic memory usage than SMKL.

We have evaluated DSRVM on challenging benchmark datasets, including the ShoulderPain, DISFA, SEMAINE, as well as on the Sonnenburg's artificial dataset. The metrics that we have used for our evaluation and comparison with RVM and SMKL are the mean squared error (MSE), and Pearson correlation coefficient (CORR). In most cases, DSRVM yields higher CORR and lower MSE than RVM and SMKL. In addition, DSRVM can be used to provide insights in the nature of facial expressions, since it learns which face parts provide the most relevant visual cues for estimating the target facial behavior.

ACKNOWLEDGMENTS

This work has been funded by the European Commission Horizon 2020 [H2020/2014-2020] under grant agreement no. 645094 (SEWA). S. Todorovic is supported in part by the National Science Foundation under grant IIS-1302700.

REFERENCES

[1] J. N. Bassili, "Facial motion in the perception of faces and of emotional expression," *J. Exp. Psychol.*, vol. 4, no. 3, pp. 373–379, 1978.

[2] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 27:1–27:27, 2011.

[3] R. Close, J. N. Wilson, and P. Gader, "A Bayesian approach to localized multi-kernel learning using the relevance vector machine," in *Geosci. Remote Sens. Symp.*, 2011, pp. 1103–1106.

[4] K. D. Craig, "The facial expression of pain Better than a thousand words?" *APS J.*, vol. 1, no. 3, pp. 153–162, 1992.

[5] K. D. Craig, K. M. Prkachin, and R. E. Grunau, "The Facial Expression of Pain," in *Handb. Pain Assess.*, 2011, pp. 117–133.

[6] T. Damoulas, Y. Ying, M. A. Girolami, and C. Campbell, "Inferring Sparse Kernel Combinations and Relevance Vectors: An Application to Subcellular Localization of Proteins," in *Int. Conf. Mach. Learn. Appl.*, 2008, pp. 577–582.

[7] F. Dornaika and J. Orozco, "Real time 3D face and facial feature tracking," *J. Real-Time Image Process.*, vol. 2, no. 1, pp. 35–44, 2007.

[8] P. Ekman, W. V. Friesen, and J. C. Hager, *Facial action coding system*. Salt Lake City, UT: A Human Face, 2002.

[9] P. Ekman and E. L. Rosenberg, *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System*. Oxford Univ. Press, USA, 2005.

[10] B. Gholami, W. Haddad, and A. Tannenbaum, "Relevance vector machine learning for neonate pain intensity assessment using digital imaging," *IEEE Trans. Biomed. Eng.*, vol. 57, no. 6, pp. 1457–1466, 2010.

[11] M. Gönen and E. Alpaydin, "Multiple kernel learning algorithms," *J. Mach. Learn. Res.*, vol. 12, pp. 2211–2268, 2011.

[12] H. Gunes and B. Schuller, "Categorical and dimensional affect analysis in continuous input: Current trends and future directions," *Image Vis. Comput.*, vol. 31, no. 2, pp. 120–136, 2013.

[13] S. D. Gunnery, J. A. Hall, and M. A. Ruben, "The Deliberate Duchenne Smile: Individual Differences in Expressive Control," *J. Nonverbal Behav.*, vol. 37, no. 1, pp. 29–41, 2013.

[14] Z. Hammal and J. F. Cohn, "Automatic detection of pain intensity," in *Proc. 14th Int. Conf. Multimodal Interact.* Santa Monica, CA, USA: ACM, oct 2012, pp. 47–52.

[15] U. Hess, R. Banse, and A. Kappas, "The intensity of facial expression is determined by underlying affective state and social situation," *J. Personal. Soc. Psychol.*, vol. 69, no. 2, pp. 280–288, 1995.

[16] U. Hess, S. Blairy, and R. E. Kleck, "The intensity of emotional facial expressions and decoding accuracy," *J. Nonverbal Behav.*, vol. 21, no. 4, pp. 241–257, 1997.

[17] L. A. Jeni, J. M. Girard, J. F. Cohn, and F. De La Torre, "Continuous AU Intensity Estimation using Localized, Sparse Facial Feature Space," *10th Int. Conf. Autom. Face Gesture Recognit.*, 2013.

[18] B. Jiang, M. F. Valstar, B. Martinez, and M. Pantic, "Dynamic Appearance Descriptor Approach to Facial Actions Temporal Modelling," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 44, no. 2, pp. 161–174, 2014.

[19] S. Kaltwang, O. Rudovic, and M. Pantic, "Continuous Pain Intensity Estimation from Facial Expressions," in *Adv. Vis. Comput.*, ser. Lecture Notes in Computer Science, vol. 7432. Heidelberg: Springer, jul 2012, pp. 368–377.

[20] S. Koelstra, M. Pantic, and I. Patras, "A Dynamic Texture-Based Approach to Recognition of Facial Actions and Their Temporal Models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 11, pp. 1940–1954, nov 2010.

[21] G. C. Littlewort, M. S. Bartlett, and K. Lee, "Automatic coding of facial expressions displayed during posed and genuine pain," *Image Vis. Comput.*, vol. 27, no. 12, pp. 1797–1803, 2009.

[22] P. Liu, J. T. Zhou, I. W.-H. Tsang, Z. Meng, S. Han, and Y. Tong, "Feature disentangling machine - a novel approach of feature selection and disentangling in facial expression analysis," in *Comput. Vision–ECCV 2014*. Springer, 2014, pp. 151–166.

[23] P. Lucey, J. F. Cohn, K. M. Prkachin, P. E. Solomon, S. Chew, and I. Matthews, "Painful monitoring: Automatic pain monitoring using the UNBC-McMaster shoulder pain expression archive database," *Image Vis. Comput.*, vol. 30, no. 3, pp. 197–205, 2012.

[24] P. Lucey, J. Cohn, K. Prkachin, P. Solomon, and I. Matthews, "Painful data: The UNBC-McMaster shoulder pain expression

archive database," in *Int. Conf. Autom. Face Gesture Recognit.* IEEE, 2011, pp. 57–64.
[25] S. Mavadati, M. Mahoor, K. Bartlett, P. Trinh, and J. F. Cohn, "DISFA: A Spontaneous Facial Action Intensity Database," *IEEE Trans. Affect. Comput.*, vol. 4, no. 2, pp. 151–160, 2013.
[26] G. McKeown, M. Valstar, R. Cowie, M. Pantic, and M. Schroder, "The SEMAINE Database: Annotated Multimodal Records of Emotionally Colored Conversations between a Person and a Limited Agent," *IEEE Trans. Affect. Comput.*, vol. 3, no. 1, pp. 5–17, 2012.
[27] L. Nanni, S. Brahnam, and A. Lumini, "A local approach based on a Local Binary Patterns variant texture descriptor for classifying pain states," *Expert Syst. Appl.*, vol. 37, no. 12, pp. 7888–7894, 2010.
[28] M. Nicolaou, V. Pavlovic, and M. Pantic, "Dynamic Probabilistic CCA for Analysis of Affective Behaviour and Fusion of Continuous Annotations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1299–1311, 2014.
[29] M. A. Nicolaou, H. Gunes, and M. Pantic, "Output-associative rvm regression for dimensional and continuous emotion prediction," *Image Vis. Comput.*, vol. 30, no. 3, pp. 186–196, 2012.
[30] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, 2002.
[31] A. Ortony and T. J. Turner, "What's basic about basic emotions?" *Psychol. Rev.*, vol. 97, no. 3, pp. 315–331, 1990.
[32] M. Pantic, "Machine analysis of facial behaviour: Naturalistic and dynamic behaviour," *Philos. Trans. R. Soc. B Biol. Sci.*, vol. 364, no. 1535, pp. 3505–3513, 2009.
[33] K. Prkachin and P. Solomon, "The structure, reliability and validity of pain expression: Evidence from patients with shoulder pain," *Pain*, vol. 139, no. 2, pp. 267–274, 2008.
[34] S. Qiu and T. Lane, "Multiple Kernel Learning for Support Vector Regression," Univ. New Mexico, Tech. Rep., 2005.
[35] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet, "SimpleMKL," *J. Mach. Learn. Res.*, vol. 9, pp. 2491–2521, 2008.
[36] P. Reicherts, A. B. M. Gerdes, P. Pauli, and M. J. Wieser, "On the mutual effects of pain and emotion," *Pain*, vol. 154, no. 6, pp. 793–800, 2013.
[37] B. Romera-Paredes, A. Argyriou, N. Berthouze, and M. Pontil, "Exploiting Unrelated Tasks in Multi-Task Learning," in *Proc. 15th Int. Conf. Artif. Intell. Stat.*, vol. 22, La Palma, Canary Islands, 2012, pp. 951–959.
[38] O. Rudovic, V. Pavlovic, and M. Pantic, "Context-sensitive Dynamic Ordinal Regression for Intensity Estimation of Facial Action Units," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 5, pp. 944–958, 2015.
[39] J. A. Russell, "Core affect and the psychological construction of emotion." *Psychol. Rev.*, vol. 110, no. 1, pp. 145–172, 2003.
[40] J. Russell, "A circumplex model of affect," *J. Personal. Soc. Psychol.*, vol. 39, no. 6, pp. 1161–1178, 1980.
[41] G. Sandbach, S. Zafeiriou, M. Pantic, and D. Rueckert, "Recognition of 3D facial expression dynamics," *Image Vis. Comput.*, vol. 30, no. 10, pp. 762–773, 2012.
[42] A. Savran, B. Sankur, and M. T. Bilge, "Regression-based intensity estimation of facial action units," *Image Vis. Comput.*, vol. 30, no. 10, pp. 774–784, 2012.
[43] B. Schuller, M. Valster, F. Eyben, R. Cowie, and M. Pantic, "AVEC 2012: the continuous audio/visual emotion challenge," in *Proc. 14th Int. Conf. Multimodal Interact.* Santa Monica, CA, USA: ACM, 2012, pp. 449–456.
[44] P. E. Shrout and J. L. Fleiss, "Intraclass correlations: uses in assessing rater reliability," *Psychol. Bull.*, vol. 86, no. 2, pp. 420–428, 1979.
[45] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Stat. Comput.*, vol. 14, no. 3, pp. 199–222, 2004.
[46] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf, "Large Scale Multiple Kernel Learning," *J. Mach. Learn. Res.*, vol. 7, pp. 1531–1565, 2006.
[47] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, 2001.
[48] M. Tipping and A. C. Faul, "Fast marginal likelihood maximisation for sparse Bayesian models," in *Proc. 9th Int. Work. Artif. Intell. Stat.*, Key West, FL, 2003, pp. 1–13.
[49] D. Tzikas, A. Likas, and N. Galatsanos, "Large scale multikernel RVM for object detection," in *Adv. Artif. Intell.* Springer, 2006, pp. 389–399.
[50] G. Tzimiropoulos and M. Pantic, "Gauss-Newton Deformable Part Models for Face Alignment in-the-Wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* IEEE, 2014, pp. 1851–1858.
[51] M. F. Valstar and M. Pantic, "Fully Automatic Recognition of the Temporal Phases of Facial Actions," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 42, no. 1, pp. 28–43, 2012.
[52] A. C. d. C. Williams, "Facial expression of pain: An evolutionary account," *Behav. Brain Sci.*, vol. 25, no. 04, pp. 439–455, 2002.
[53] S. Zafeiriou and I. Pitas, "Discriminant Graph Structures for Facial Expression Recognition," *IEEE Trans. Multimed.*, vol. 10, no. 8, pp. 1528–1540, 2008.
[54] Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang, "A survey of affect recognition methods: audio, visual, and spontaneous expressions," *IEEE Trans. Patt. Anal. Mach. Intel.*, vol. 31, no. 1, pp. 39–58, 2009.
[55] G. Zhao and M. Pietikainen, "Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 915–928, 2007.
[56] K. Zhao, W.-S. Chu, F. la Torre, J. F. Cohn, and H. Zhang, "Joint Patch and Multi-Label Learning for Facial Action Unit Detection," in *IEEE Conf. Comput. Vis. Pattern Recognit.*, jun 2015.
[57] L. Zhong, Q. Liu, P. Yang, B. Liu, J. Huang, and D. N. Metaxas, "Learning active facial patches for expression analysis," in *IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 2562–2569.
[58] J. Zhu and E. P. Xing, "On Primal and Dual Sparsity of Markov Networks," in *Int. Conf. Mach. Learn.*, Montreal, QC, Canada, 2009, pp. 1265–1272.

Sebastian Kaltwang received the masters degree in computer science (German "Diplom-Informatik") from the Karlsruhe Institute of Technology, Germany in 2011. Recently, he obtained the PhD degree within the Department of Computing at Imperial College London, United Kingdom. His research interests include automatic human behavior analysis, machine learning and computer vision.

Sinisa Todorovic received the PhD degree in electrical and computer engineering from the University of Florida in 2005. He is an assistant professor in the School of Electrical Engineering and Computer Science at Oregon State University. He was a postdoctoral research associate in the Beckman Institute at the University of Illinois at Urbana-Champaign, between 2005 and 2008. His research interests include computer vision and machine learning problems. He is a member of the IEEE.

Maja Pantic is a professor in affective and behavioral computing in the Department of Computing at Imperial College London, United Kingdom, and in the Department of Computer Science at the University of Twente, The Netherlands. She currently serves as the editor in chief of Image and Vision Computing Journal and as an associate editor for both the IEEE Transactions on Systems, Man, and Cybernetics Part B and the IEEE Transactions on Pattern Analysis and Machine Intelligence. She has received various awards for her work on automatic analysis of human behavior, including the European Research Council Starting Grant Fellowship 2008 and the Roger Needham Award 2011. She is a fellow of the IEEE.

Supplementary Material Doubly Sparse Relevance Vector Machine for Continuous Facial Behavior Estimation

Sebastian Kaltwang, Sinisa Todorovic, *Member, IEEE* and Maja Pantic, *Fellow, IEEE*

Abstract—Certain inner feelings and physiological states like pain are subjective states that cannot be directly measured, but can be estimated from spontaneous facial expressions. Since they are typically characterized by subtle movements of facial parts, analysis of the facial details is required. To this end, we formulate a new regression method for continuous estimation of the intensity of facial behavior interpretation, called Doubly Sparse Relevance Vector Machine (DSRVM). DSRVM enforces double sparsity by jointly selecting the most relevant training examples (a.k.a. relevance vectors) and the most important kernels associated with facial parts relevant for interpretation of observed facial expressions. This advances prior work on multi-kernel learning, where sparsity of relevant kernels is typically ignored. Empirical evaluation on challenging Shoulder Pain videos, and the benchmark DISFA and SEMAINE datasets demonstrate that DSRVM outperforms competing approaches with a multi-fold reduction of running times in training and testing.

Index Terms—Regression, Relevance Vector Machine, Multiple Kernel Learning, Facial expressions

APPENDIX A
SEMAINE SESSION NUMBERS

To allow researchers to reproduce results on the same subset of SEMAINE, we provide here the session numbers used in our experiments:

19, 20, 21, 22, 29, 30, 31, 34, 35, 36, 37, 40, 41, 42, 43, 46, 47, 48, 49, 60, 70, 71, 72, 73, 76, 77, 78, 79, 82, 83, 84, 85, 94, 95, 96, 97, 106, 107, 108, 109, 113, 114, 115

APPENDIX B
DSRVM UPDATE FORMULAS

In the following, we derive the update formulas for each step of the DSRVM algorithm:

Step 1: In order to derive a variational update formula for $q(\mathbf{w})$, we need to solve the expectation of the joint distribution with respect to $q(\mathbf{v})$, since $\log q^\star(\mathbf{w}) \sim \mathbb{E}_{q(\mathbf{v})}\left[\log p(\mathbf{w},\mathbf{v},\mathbf{t}|\boldsymbol{\alpha}_*,\boldsymbol{\beta}_*,\sigma_*^2)\right]$. This leads to the optimal solution:

$$\begin{aligned} &q^\star(\mathbf{w}) \sim \mathcal{N}(\mathbf{w};\boldsymbol{\mu},\boldsymbol{\Sigma}), \\ &\quad \boldsymbol{\mu} = \sigma^{-2}\boldsymbol{\Sigma}\left(\textstyle\sum_k \mathbb{E}\left[v_k\right]\boldsymbol{\Phi}_k^\top\right)\mathbf{t}, \\ &\quad \boldsymbol{\Sigma} = \left(\mathbf{A} + \sigma^{-2}\textstyle\sum_{k_1}\sum_{k_2}\mathbb{E}\left[v_{k_1}v_{k_2}\right]\boldsymbol{\Phi}_{k_1}^\top\boldsymbol{\Phi}_{k_2}\right)^{-1}, \end{aligned} \tag{19}$$

where $\mathbb{E}[\cdot]$ is the expectation using the posterior distribution in (7), and $\boldsymbol{\Phi}_k \in \mathbb{R}^{N\times M}$ with $\boldsymbol{\Phi}_k(n,m) = \kappa_k(\mathbf{x}_n,\mathbf{x}_m)$. Since $q(\mathbf{v})$ is Gaussian as well, computation of $\mathbb{E}\left[v_k\right]$ and $\mathbb{E}\left[v_{k1}v_{k2}\right]$ is straightforward.

Step 2: Analogously, we derive the update formula for $q(\mathbf{v})$:

$$\begin{aligned} &q^\star(\mathbf{v}) \sim \mathcal{N}(\mathbf{v};\boldsymbol{\nu},\boldsymbol{\Lambda}), \\ &\quad \boldsymbol{\nu} = \sigma^{-2}\boldsymbol{\Lambda}\left(\textstyle\sum_m \mathbb{E}\left[w_m\right]\boldsymbol{\Psi}_m^\top\right)\mathbf{t}, \\ &\quad \boldsymbol{\Lambda} = \left(\mathbf{B} + \sigma^{-2}\textstyle\sum_{m_1}\sum_{m_2}\mathbb{E}\left[w_{m_1}w_{m_2}\right]\boldsymbol{\Psi}_{m_1}^\top\boldsymbol{\Psi}_{m_2}\right)^{-1}, \end{aligned} \tag{20}$$

where $\boldsymbol{\Psi}_m \in \mathbb{R}^{N\times K}$ with $\boldsymbol{\Psi}_m(n,k) = \kappa_k(\mathbf{x}_n,\mathbf{x}_m)$.

Step 3: In order to get an efficient update rule for $\boldsymbol{\alpha}_*$, we further approximate $q(\mathbf{v})$ with a delta function at its mode $\boldsymbol{\nu}$. Thus, the marginal likelihood is approximated by:

$$p(\mathbf{t}|\boldsymbol{\alpha}_*,\boldsymbol{\beta}_*,\sigma_*^2) \approx p(\mathbf{t}|\mathbf{v},\boldsymbol{\alpha}_*,\sigma_*^2)\delta(\mathbf{v}). \tag{21}$$

Taking into account a uniform prior for $\boldsymbol{\alpha}$ and σ^2, and following the same update formula as for the original RVM [47], the solution is a convolution of Gaussians and can be expressed in closed form

$$p(\mathbf{t}|\mathbf{v},\boldsymbol{\alpha}_*,\sigma_*^2) \sim \mathcal{N}(\mathbf{t};\mathbf{0},\mathbf{C}_\mathbf{v}), \tag{22}$$

where $\mathbf{C}_\mathbf{v} = \boldsymbol{\Phi}_\mathbf{v}\mathbf{A}^{-1}\boldsymbol{\Phi}_\mathbf{v}^\top + \sigma^2\mathbf{I}$ and $\boldsymbol{\Phi}_\mathbf{v} \in \mathbb{R}^{N\times M}$ with $\boldsymbol{\Phi}_\mathbf{v}(n,m) = \sum_k v_k\kappa_k(\mathbf{x}_n,\mathbf{x}_m)$. As described in [48], we can derive an optimal update for each α_m separately as

$$\alpha_m^* = \begin{cases} \frac{a_m^2}{b_m^2 - a_m} & \text{if } b_m^2 > a_m \\ \infty & \text{otherwise} \end{cases}, \tag{23}$$

where $a_m = \phi_m^\top\mathbf{C}_{-m}^{-1}\phi_m$, $b_m = \phi_m^\top\mathbf{C}_{-m}^{-1}\mathbf{t}$, $\mathbf{C}_{-m} = \sigma^2\mathbf{I} + \sum_{i\neq m}\alpha_i^{-1}\phi_i\phi_i^\top$ and ϕ_m is the mth column of $\boldsymbol{\Phi}_\mathbf{v}$ with $\phi_m(n) = \sum_k v_k\kappa_k(\mathbf{x}_n,\mathbf{x}_m)$. Note that in each iteration only α_m with the largest likelihood increase is updated, for details see [48]. Setting α_m to infinite effectively prunes

- *S. Kaltwang and M. Pantic are with the Department of Computing, Imperial College London, 180 Queens Gate, London SW7 2AZ, UK. M. Pantic is also with the EEMCS, University of Twente, the Netherlands. E-mail: {sk2608, m.pantic}@imperial.ac.uk.*
- *S. Todorovic is with the School of EECS, Oregon State University, 1148 Kelley Engineering Center, Corvallis, OR 97331. E-mail: sinisa@eecs.oregonstate.edu.*

out the corresponding basis function ϕ_m, i.e. only the basis with $\alpha_m < \infty$ are used for inference.

Step 4: For deriving an update formula for $\boldsymbol{\beta}_*$, we follow the same reasoning as in step 3. We approximate $q(\mathbf{w})$ with a delta function at its mode $\boldsymbol{\mu}$. Then, we approximate the marginal likelihood as

$$p(\mathbf{t}|\boldsymbol{\alpha}_*, \boldsymbol{\beta}_*, \sigma_*^2) \approx p(\mathbf{t}|\mathbf{w}, \boldsymbol{\beta}_*, \sigma_*^2)\delta(\mathbf{w}). \tag{24}$$

and analogously to step 3, we need to maximize

$$p(\mathbf{t}|\mathbf{w}, \boldsymbol{\beta}_*, \sigma_*^2) \sim \mathcal{N}(\mathbf{t}; \mathbf{0}, \mathbf{D}_\mathbf{w}), \tag{25}$$

where $\mathbf{D}_\mathbf{w} = \boldsymbol{\Psi}_\mathbf{w}\mathbf{B}^{-1}\boldsymbol{\Psi}_\mathbf{w}^\top + \sigma^2\mathbf{I}$ and $\boldsymbol{\Psi}_\mathbf{w} \in \mathbb{R}^{N\times K}$ with $\boldsymbol{\Psi}_\mathbf{w}(n,k) = \sum_m w_m \kappa_k(\mathbf{x}_n, \mathbf{x}_m)$. As above, we can derive an optimal update for each β_k separately as

$$\beta_k^* = \begin{cases} \frac{c_k^2}{d_k^2 - c_k} & \text{if } d_k^2 > c_k \\ \infty & \text{otherwise} \end{cases}, \tag{26}$$

where $c_k = \psi_k^\top \mathbf{D}_{-k}^{-1}\psi_k$, $d_k = \psi_k^\top \mathbf{D}_{-k}^{-1}\mathbf{t}$, $\mathbf{D}_{-k} = \sigma^2\mathbf{I} + \sum_{i\neq k}\beta_i^{-1}\psi_i\psi_i^\top$ and ψ_k is the kth column of $\boldsymbol{\Psi}_\mathbf{w}$ with $\psi_k(n) = \sum_m w_m \kappa_k(\mathbf{x}_n, \mathbf{x}_m)$. Again, note that only β_k with the largest likelihood increase is updated in each iteration.

Step 5: We further derive an update for the noise variance σ_*^2, by solving the derivative regarding the marginal likelihood $\frac{\partial \mathcal{L}}{\partial \sigma^2} = 0$. This leads to the formula:

$$\sigma_*^2 = \tfrac{1}{N}\mathbb{E}_{\mathbf{w},\mathbf{v}}\left[\|\mathbf{t} - y(\mathbf{X}; \mathbf{w}, \mathbf{v})\|^2\right], \tag{27}$$

$$\mathbb{E}_{\mathbf{w},\mathbf{v}}\left[\|\mathbf{t} - y(\mathbf{X}; \mathbf{w}, \mathbf{v})\|^2\right] = \|\mathbf{t} - y(\mathbf{X}; \mathbb{E}[\mathbf{w}], \mathbb{E}[\mathbf{v}])\|^2 + \sum_{m_1,m_2,k_1,k_2} \boldsymbol{\Sigma}(m_1,m_2)\boldsymbol{\Lambda}(k_1,k_2)\sum_n \kappa_{n,m_1,k_1}\kappa_{n,m_2,k_2}$$

with $y(\mathbf{X}; \mathbf{w}, \mathbf{v}) = [y(\mathbf{x}_n; \mathbf{w}, \mathbf{v})]_{n=1}^N$ and $\kappa_{n,m,k} = \kappa_k(\mathbf{x}_n, \mathbf{x}_m)$.

APPENDIX C
RVM UPDATE FORMULAS

In the following, we describe each of the RVM steps in detail and compare it with our DSRVM update steps.

Step a: $p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}_*, \sigma_*^2)$ is a convolution of two Gaussians and can hence be calculated in closed form:

$$\begin{aligned} p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}_*, \sigma_*^2) &\sim \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma}), \\ \boldsymbol{\mu} &= \sigma^{-2}\boldsymbol{\Sigma}\boldsymbol{\Phi}^\top\mathbf{t}, \\ \boldsymbol{\Sigma} &= \left(\mathbf{A} + \sigma^{-2}\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\right)^{-1}, \end{aligned} \tag{28}$$

where $\boldsymbol{\Phi}$ is the kernel basis matrix with $\boldsymbol{\Phi}(n,m) = \kappa(\mathbf{x}_n, \mathbf{x}_m)$. It follows that RVM uses a linear function of the targets to estimate its parameters $\mathbf{w}_{\text{RVM}} = (\sigma^{-2}\boldsymbol{\Sigma}\boldsymbol{\Phi}^\top)\mathbf{t} = L\mathbf{t}$. While the convexity and closed-form of such linear RVM formulation is appealing, the use of the linear function strongly restricts the complexity of data that RVM can represent. By contrast, our DSRVM introduces kernel weights $\mathbf{v}$ that play a role of hidden variables in the estimation of $\mathbf{w}_{\text{DSRVM}} = \sigma^{-2}\boldsymbol{\Sigma}_\mathbf{v}\boldsymbol{\Phi}_\mathbf{v}^\top\mathbf{t}$ (see (10)), where each configuration of $\mathbf{v}$ values corresponds to a particular component in the exponentially large mixture of distributions of $\mathbf{w}_{\text{DSRVM}}$. This significantly extends the modeling capacity of our DSRVM relative to that of RVM.

Step b: The marginal likelihood $\mathcal{L}_{\text{RVM}}$ can be maximized regarding $\boldsymbol{\alpha}_*$ by the same update formula as in (23), except that $\phi_m(n) = \kappa(\mathbf{x}_n, \mathbf{x}_m)$ due to the single fixed kernel.

Step c: Optimizing σ_*^2 by taking the corresponding derivative of $\mathcal{L}_{\text{RVM}}$ leads to the update formula:

$$\sigma_*^2 = \frac{\|\mathbf{t} - y_{\text{RVM}}(\mathbf{X}; \mathbb{E}[\mathbf{w}])\|^2}{N - M + \sum_m \alpha_m \boldsymbol{\Sigma}(m,m)}. \tag{29}$$

The DSRVM σ_*^2 update (27) has a different form than (29), because the DSRVM target $\mathcal{L}$ is a variational approximation while $\mathcal{L}_{\text{RVM}}$ is Gaussian.

APPENDIX D
ICC RESULTS

In order to compare to [25], we also provide the results for the Intra-Class Correlation Coefficient ICC(3,1) [44] performance measure. Tab. 5 shows the ICC results on the ShoulderPain and SEMAINE data. DSRVM performs best in most of the cases, with some exceptions where it is on par with RVM-all and RVM-sep.

TABLE 5: Results on the ShoulderPain data for the pain targets (left column) and on the SEMAINE data for the arousal (Ar.) and valence (Val.) targets (right column). DSRVM is compared to different RVM and SMKL. For ShoulderPain, video features are extracted at different space-time (S-T) scales. p-value measures significance of the result in comparison to DSRVM. The table shows Intra-class Correlation Coefficient (ICC). The best results are marked bold. If two results differ by at most 0.01 and the p-value is greater than 0.05, we mark both results bold.

		ShoulderPain						SEMAINE	
	Method	S6x6 T1	S6x6 T10	S6x6 T20	S9x9 T1	S9x9 T10	S9x9 T20	Ar.	Val.
ICC	DSRVM	**0.52**	**0.53**	**0.47**	**0.49**	**0.48**	**0.46**	**0.21**	**0.20**
	RVM all	0.50	0.51	**0.48**	**0.48**	0.45	0.44	**0.20**	0.17
	p-value	.75	.89	.63	.20	.46	.26	.45	.24
	RVM best	0.15	0.34	0.38	0.32	0.40	0.37	0.14	0.13
	p-value	.00	.00	.13	.00	.01	.00	.10	.39
	RVM sep	**0.53**	0.51	0.49	0.47	0.43	0.43	0.18	0.19
	p-value	.59	.85	.49	.55	.26	.96	.37	.36
	SMKL	0.48	0.49	0.46	0.47	0.46	0.44	0.18	0.16
	p-value	.03	.06	.07	.13	.10	.07	.11	.07
	mRVM	0.40	0.40	0.36	0.37	0.40	0.41	0.05	0.05
	p-value	.00	.01	.12	.22	.45	.81	.00	.00

Tab. 6 shows the ICC results on the DISFA data. DSRVM is on par with RVM-sep, each of the methods is the best for half of the FAU targets. On average they perform similar, followed by RVM-all and SMKL. Although DSRVM and RVM-best perform similar regarding ICC, DSRVM selects less RV and the testing time is about 8 times faster, as can be seen in Tab. 3.

Tab. 7 shows the ICC results on the DISFA data while using the same evaluation procedure as in [25]. The first step is to train a SR subspace [**?**] with data from all

subjects and thus the results are subject-dependent. Then the DSRVM model is trained on 3000 samples with features from the learned subspace and a single Gaussian kernel, while using a leave-one-subject-out cross-validation procedure. Thus the combined method is shown as SR+DSRVM. The results show show a better DSRVM performance in the majority of cases, including the average of all AUs (AVG).

APPENDIX E
COMPARISON WITH CLASSIFICATION

To demonstrate the advantage of continuous regression models over classifiers, we compare the continuous DSRVM with the discrete mRVM [6] on the artificial and the SEMAINE dataset. Fig. 7 (top left) shows the mRVM results for discretizing the targets into different numbers of classes on the artificial data. The DSRVM results are plotted for comparison. Note that they are obtained directly for continuous-valued targets, and thus appear as constant in these plots across different settings of discretization of target values. We see that the optimum result for mRVM is reached at 8 classes, but DSRVM results in superior performance at all times.

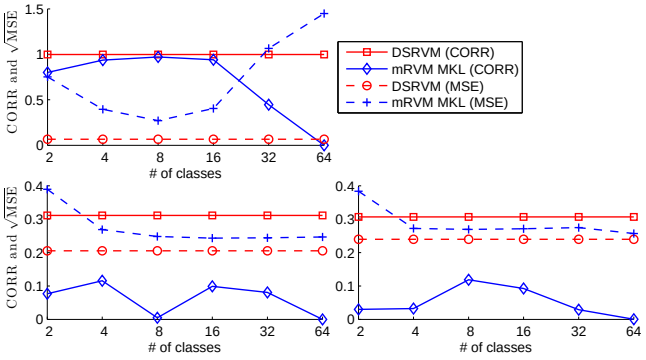

Fig. 7: Results on the artificial data (top left) and on the SEMAINE data for the arousal (bottom left) and the valence (bottom right) target: Comparison of the continuous DSRVM with the discrete mRVM method [6] by discretizing the targets into different number of classes. (Note: we use the square root of the MSE for better axis scaling)

As can be seen from Fig. 7 (bottom left and right), we have a similar case for the SEMAINE data. Again DSRVM outperforms mRVM, independently of the number of classes the targets are divided into. The MSE becomes constant as the number of classes increases, since the mRVM cannot train properly due to too few data per class and therefore assigns the majority class to all instances. Due to the bias in the target distribution, this leads to a low MSE since most instances are close to the majority value around 0. However, the CORR as a relative measure takes the variance of predictions into account and clearly states that the target is modeled badly.

TABLE 8: Model statistics on the DISFA data averaged over all FAU targets. Shown are the number of relevance vectors (#RV), the number of relevant kernels (#RK), training runtime (TRN) in sec$\times 10^2$, and test runtime (TST) in sec. #RK is only shown for models that adjust the kernel.

	Method		#RV	#RK	TRN	TST
MSE	DSRVM		31.6	13.4	21.3	0.3
	RVM all	[47]	102.8	-	1.2	1.3
	RVM best	[47]	65.4	-	1.3	0.1
	RVM sep	[3], [49]	199.9	-	36.2	3.1
	SMKL	[35]	1913.6	33.1	149.8	7.8
	mRVM	[6]	37.4	36.0	77.4	1.2
	SR+SVM	[25]	464.2	-	0.1	0.1
ICC	DSRVM		55.5	21.4	21.3	0.4
	RVM all	[47]	120.6	-	1.2	1.3
	RVM best	[47]	77.7	-	1.3	0.1
	RVM sep	[3], [49]	217.6	-	36.2	3.1
	SMKL	[35]	1914.2	32.5	149.8	7.8
	mRVM	[6]	51.4	36.0	77.4	1.2
	SR+SVM	[25]	464.2	-	0.1	0.1

APPENDIX F
MODEL STATISTICS FOR MSE AND ICC

Tab. 8 provides average statistics of the learned models optimized for MSE and ICC, which are similar to the CORR statistics in Tab. 3. #RV and #RK tend to be a bit lower for MSE than for CORR and ICC. This is probably caused by the imbalance of the data, since MSE favors simpler models (with fewer #RV and #RK) that conservatively rate intensity zero.

TABLE 6: Results on the DISFA data for different FAU targets. See the caption of Tab. 5. The last column shows the average results over all FAUs (AVG).

	Method	AU1	AU2	AU4	AU5	AU6	AU9	AU12	AU15	AU17	AU20	AU25	AU26	AVG
ICC	DSRVM	0.26	0.22	**0.47**	0.14	**0.47**	**0.40**	**0.75**	0.28	0.34	**0.19**	**0.58**	0.35	**0.37**
	RVM all	0.24	**0.27**	0.34	0.10	0.42	0.30	0.72	0.30	**0.38**	0.17	**0.59**	0.36	0.35
	p-value	.25	.09	.00	.06	.30	.00	.24	.41	.22	.34	.94	.51	.17
	RVM best	0.17	0.14	0.41	**0.16**	0.37	0.29	0.69	0.15	0.19	0.14	0.56	0.14	0.29
	p-value	.06	.06	.24	.15	.21	.07	.04	.01	.00	.07	.52	.01	.00
	RVM sep	**0.33**	0.24	0.45	0.14	0.44	0.31	**0.74**	**0.31**	0.37	**0.19**	**0.59**	**0.38**	**0.37**
	p-value	.22	.19	.62	.65	.46	.03	.54	.06	.40	.67	.73	.29	.74
	SMKL	0.30	0.21	0.44	0.12	0.41	0.29	0.73	0.28	0.34	0.15	0.58	0.34	0.35
	p-value	.56	.88	.40	.03	.26	.02	.30	.91	.94	.08	.85	.77	.05
	mRVM	0.19	0.15	0.24	0.04	0.35	0.38	0.65	0.19	0.27	**0.19**	0.57	0.16	0.28
	p-value	.40	.40	.05	.04	.04	.59	.01	.19	.23	.88	.64	.01	.00
	SR+SVM	0.10	0.10	0.30	0.03	0.37	0.36	0.74	0.18	0.27	0.15	0.46	0.33	0.28
	p-value	.01	.02	.00	.03	.07	.27	.70	.09	.19	.16	.03	.44	.00

TABLE 7: ICC results on the DISFA data for different FAU targets. SR+DSRVM is compared to SR+SVM within a subject-dependent setting that corresponds to the same evaluation procedure as in [25]. The last column shows the average results over all FAUs (AVG).

Method	AU1	AU2	AU4	AU5	AU6	AU9	AU12	AU15	AU17	AU20	AU25	AU26	AVG
SR+DSRVM	**0.83**	0.81	**0.87**	**0.69**	**0.83**	**0.83**	**0.88**	**0.80**	**0.79**	**0.73**	0.89	0.75	**0.81**
SR+SVM [25]	0.80	**0.83**	**0.87**	0.58	0.81	0.80	0.84	0.71	0.69	0.54	**0.94**	**0.79**	0.77