Latent Trees for Estimating Intensity of Facial Action Units

Sebastian Kaltwang
Imperial College London
sk2608@imperial.ac.uk

Sinisa Todorovic
Oregon State University
sinisa@eecs.oregonstate.edu

Maja Pantic
Imperial College London
m.pantic@imperial.ac.uk

Abstract

This paper is about estimating intensity levels of Facial Action Units (FAUs) in videos as an important step toward interpreting facial expressions. As input features, we use locations of facial landmark points detected in video frames. To address uncertainty of input, we formulate a generative latent tree (LT) model, its inference, and novel algorithms for efficient learning of both LT parameters and structure. Our structure learning iteratively builds LT by adding either a new edge or a new hidden node to LT, starting from initially independent nodes of observable features. A graph-edit operation that increases maximally the likelihood and minimally the model complexity is selected as optimal in each iteration. For FAU intensity estimation, we derive closed-form expressions of posterior marginals of all variables in LT, and specify an efficient bottom-up/top-down inference. Our evaluation on the benchmark DISFA and ShoulderPain datasets, in subject-independent setting, demonstrate that we outperform the state of the art, even under significant noise in facial landmarks. Effectiveness of our structure learning is demonstrated by probabilistically sampling meaningful facial expressions from the LT.

1. Introduction

Analyzing spontaneous facial expressions in video is important for a wide range of applications in human-computer and human-human interactions. This is because spontaneous expressions are an integral part of human communication, e.g., capable of conveying various attitudes and emotions (e.g., spontaneous frowns), and revealing certain thoughts (e.g., spontaneous eye squints and brow scowls) [5]. While their detection in terms of presence or absence in video has already gained traction in the literature (e.g., [27, 11]), there is relatively scant work on estimating their intensities [21, 18, 22, 12]. Yet, the meaning and function of spontaneous facial expressions depends largely on their intensity [9, 10, 17]. For example, smiles of enjoyment are typically full-blown smiles [7], whereas smiles of faked happiness or sarcasm are usually weaker in intensity when observed in natural social settings.

In this paper, we address the problem of identifying intensity levels of Facial Action Units (FAUs) as an important step toward interpreting facial expressions. Our research is motivated by findings of componential facial emotion theory [20, 23], which suggests that in spontaneous expressions (and actions) only certain parts of faces are universally displayed, and thus could be more reliably used for inferring facial expressions than entire faces. To this end, we use the facial action coding system (FACS) [4] that defines 32 FAUs as atomic, spatially and temporally well-contained facial movements, associated with five intensity levels (A<B<C<D<E). Thus, given a video frame, our goal is estimate active FAUs and score them with one of the five intensity levels.

This problem is challenging. Spontaneous expressions are generally characterized by subtle, minimal facial movements and large out-of-plane head movements [16]. Both are very difficult to track, leading to higher error rates in FAU inference. Also, certain FAU intensity levels could be modulated by the social environment or inner human experiences (e.g., chronic vs. acute pain) [26]. This makes subject-independent FAU intensity estimation difficult.

1.1. Overview of Approach and Contributions

To address the aforementioned challenges, we consider a Bayesian generative framework. We formalize our problem as that of jointly predicting multiple FAU targets, $\mathbf{T} = \{\mathbf{x}_1, ..., \mathbf{x}_T\}$, given a set of image features, $\mathbf{F} = \{\mathbf{x}_{T+1}, ..., \mathbf{x}_{T+F}\}$. Every target $\mathbf{x}_m \in \mathbf{T}$ can be defined as a vector of various attributes associated with mth FAU, and in a special case for our problem as FAU intensities. Image features $\mathbf{x}_m \in \mathbf{F}$ are defined as local descriptors of the face, which can be appearance based (e.g., patches) or locations of facial landmarks detected in a video frame.

We specify a graphical model for representing the joint distribution of targets and features, $p(\mathbf{T}, \mathbf{F})$, and use the Bayes' rule to derive an elegant solution to FAU intensity

estimation as

$$\hat{\mathbf{T}} = \max_{\mathbf{T}} \frac{p(\mathbf{T}, \mathbf{F})}{\sum_{\mathbf{T}'} p(\mathbf{T}', \mathbf{F})}. \quad [1] \tag{1}$$

Our formulation has a number of advantages over existing approaches [21, 18, 22, 12]. They typically adopt the discriminative framework for directly predicting FAU intensities given the features, e.g., using Support Vector Classification (SVC) [18], Relevance Vector Machine (RVM) [12], AdaBoost [25], or ordinal Conditional Random Fields (CRF) [21]. While discriminative approaches are generally robust, we experimentally demonstrate in this paper that they underperform under the aforementioned challenges. In particular, due to frequent partial occlusions of the face or large out-of-plane head movements in non-staged video, some input features might be missing or very unreliable. Our results show that our model can robustly handle missing input features by marginalizing them out, unlike the competing discriminative approaches. Also, our model is less likely to overfit to training human subjects, due to the joint modeling of all FAUs $\mathbf{T}$ and features $\mathbf{F}$.

For effectively capturing statistical dependencies among $\mathbf{T}$ and $\mathbf{F}$, our model has hidden (latent) random variables. Also, for ensuring modelling efficiency (e.g., few model parameters) and efficient inference of $\hat{\mathbf{T}}$, we organize the hidden variables in a tree structure, and hence call our model Latent Tree (LT). In LT, leaf nodes represent $\mathbf{T}$ and $\mathbf{F}$, and all other nodes correspond to the hidden variables (also called hidden nodes). Importantly, no other restrictions are placed on the model structure beyond the tree structure, defined by the total number of hidden nodes and edges.

LT structure is unknown a priori. We specify a new algorithm for efficient learning of both model parameters and model structure on training data. Our structure learning iteratively builds LT by introducing either new parent nodes or new connections between existing hidden nodes, depending on the resulting increase in the joint likelihood $p(\mathbf{T}, \mathbf{F})$. Our key contribution here is a heuristic algorithm for efficiently computing the maximum likelihood increase.

For FAU intensity estimation, we derive closed-form expressions of posterior marginals of all variables in LT, and specify an efficient inference of $\hat{\mathbf{T}}$ given $\mathbf{F}$ in two passes – bottom-up and top-down.

We have evaluated LT on the benchmark DISFA [18] and ShoulderPain [16] datasets, which provide per-frame FAU intensity labels for spontaneous facial expressions. For evaluation, we have used the subject-independent setting, where different human subjects appear in training and testing, as a widely adopted practice in vision and broader. This has prevented direct comparison with certain prior work which uses the same human subjects for both training and testing. In comparison with baselines and the state-of-the-art methods that also use the subject-independent setting, the results demonstrate our superior performance, even under significant noise introduced to facial landmark points. We also demonstrate effectiveness of our structure learning by probabilistically sampling locations of facial landmark points, conditioned on a given FAU intensity. Our generative sampling produces plausible facial expressions.

[1] We always use the sum symbol for marginalization, even for continuous variables, for simplicity.

1.2. Closely Related Work

The literature abounds with various formulations of generative models and their structure learning [13]. The two unique aspects of our approach, suited to FAU intensity estimation (see Sec. 1.1), include tying latent FAU intensities and observable features together at the leaf level of LT, and a novel formulation of efficient graph edits for structure learning based on the Bayesian structural Expectation-Maximization (EM) [6].

Recent work on the Binary Latent Tree (BLT) [8] puts the restrictive constraint on the model structure that every non-leaf node cannot have more than two children. Our structure learning is more efficient, and significantly differs from the way they build BLT as trading-off the Mutual Information score and the Bayesian Information Criterion (BIC). We experimentally demonstrate that their binary-tree restriction leads to poor (BLT) performance in our domain.

Structural learning of latent trees can also be formulated by grouping random variables according to their information distance [2]. However, the space of possible grouping combinations is very large, which leads to an inefficient algorithm.

In the following, Sec. 2 specifies LT; Sec. 3 formulates our inference; Sec. 4.1 presents our model parameter learning; Sec. 4.2 specifies our model structure learning; and Sec. 5 presents our results.

2. Latent Tree Formulation

This section specifies our LT for modeling $p(\mathbf{T}, \mathbf{F})$, where $\mathbf{T}$ and $\mathbf{F}$ are introduced in Sec. 1.1. Let $\mathbf{X} = \{\mathbf{T}, \mathbf{F}\} = \{\mathbf{x}_1, ..., \mathbf{x}_M\}$, $M = T + F$. To model $p(\mathbf{X})$, we use a tree that includes, in addition to $\mathbf{X}$, also L hidden discrete variables $\mathbf{H} = \{\mathbf{h}_1, ..., \mathbf{h}_L\}$, each with the same number of states K. The tree is aimed at efficiently representing joint distributions of various subsets of $\mathbf{X}$ as follows. Leaf nodes of the tree correspond to every $\mathbf{x}_m \in \mathbf{X}$, and nodes at levels closer to the root correspond to every $\mathbf{h}_l \in \mathbf{H}$. The nodes are hierarchically connected in the tree to represent that the distribution of every node $\mathbf{x}_m \in \mathbf{X}$ (or $\mathbf{h}_l \in \mathbf{H}$) is conditioned on its parent node in the tree $\mathbf{h}_{P(m)} \in \mathbf{H}$ (or $\mathbf{h}_{P(l)} \in \mathbf{H}$). Thus, the tree structure is defined by the function $P(\cdot)$ which assigns the parent to each node, or the empty set $\emptyset$ if the node is a root. A non-leaf node in the tree may have arbitrary many children nodes.

The conditional distribution between hidden nodes $\mathbf{h}_l$ and $\mathbf{h}_{P(l)}$ is categorical, since both nodes are discrete:

$$p(\mathbf{h}_l|\mathbf{h}_{P(l)}=k) = \text{Cat}(\mathbf{h}_l;\boldsymbol{\mu}_{k,l}), \tag{2}$$

where $k \in \{1,...,K\}$, $\text{Cat}(\mathbf{h};\boldsymbol{\mu})$ is the categorical distribution over $\mathbf{h} \in \{1,...,K\}$ with the parameter $\boldsymbol{\mu} \in \mathbb{R}^K$, $\forall k : \boldsymbol{\mu}_k \geq 0$, and $\sum_{k=1}^K \boldsymbol{\mu}_k = 1$. The annotated FAU targets are discrete and thus the conditional distribution between a target $\mathbf{x}_m$ and its parent $\mathbf{h}_{P(m)}$ is categorical as well, i.e. equivalent as in (2).

The conditional distribution for continuous features $\mathbf{x}_m^{(\text{cont.})}$ is Gaussian:

$$p(\mathbf{x}_m|\mathbf{h}_{P(m)}=k) = \mathcal{N}(\mathbf{x}_m;\boldsymbol{\mu}_{k,m},\boldsymbol{\Sigma}_{k,m}), \tag{3}$$

with the mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^d_+ \times \mathbb{R}^d_+$, where d is the dimensionality of $\mathbf{x}_m$. The tree root $\mathbf{h}_r$ has no parent, and thus is not conditioned on another node. Its distribution is defined as a prior:

$$p(\mathbf{h}_r|\mathbf{h}_{P(r)}) = p(\mathbf{h}_r|\emptyset) = \text{Cat}(\mathbf{h}_r;\boldsymbol{\mu}_r). \tag{4}$$

From (3)–(4), the joint distribution of all variables can be expressed as

$$p(\mathbf{X},\mathbf{H}) = \textstyle\prod_{m,l} p(\mathbf{x}_m|\mathbf{h}_{P(m)})p(\mathbf{h}_l|\mathbf{h}_{P(l)}). \tag{5}$$

We use (5) to define the marginal log-likelihood of a given set of data points $\{\mathbf{X}^{(1)},...,\mathbf{X}^{(N)}\}$ as

$$\mathcal{L} = \textstyle\sum_{n=1}^N \ln \sum_{\mathbf{H}} p(\mathbf{X}^{(n)},\mathbf{H}). \tag{6}$$

To learn LT parameters and structure, we maximize $\mathcal{L}$, given by (6), on training data using an EM algorithm. As inference is an integral part of learning, in the sequel, we first specify our inference in Sec. 3, and then present our learning of LT parameters in Sec. 4.1 and LT structure in Sec. 4.2.

3. Bottom-up/Top-down Inference on LT

We use the MAP criterion, given by (1), to predict discrete FAU intensities $\hat{\mathbf{T}} = \{\hat{\mathbf{x}}_m : m = 1,...,T\}$, given all input features $\mathbf{F}$. From our specification of LT, presented in Sec. 2, the MAP estimation of (1) can be decomposed for every individual target $\mathbf{x}_m \in \mathbf{T}$ as

$$\hat{\mathbf{x}}_m = \text{max}_{\mathbf{x}_m} \textstyle\sum_{\mathbf{h}_{P(m)}} p(\mathbf{x}_m|\mathbf{h}_{P(m)})p(\mathbf{h}_{P(m)}|\mathbf{F}). \tag{7}$$

From (7), our inference problem amounts to finding the posterior $p(\mathbf{h}_{P(m)}|\mathbf{F})$.

In the following, we explain how to compute the marginal posteriors $p(\mathbf{h}_l|\mathbf{S})$ for all hidden nodes $\mathbf{h}_l \in \mathbf{H}$ using the standard bottom-up/top-down inference (a.k.a., the inside-outside algorithm) on trees [13], where $\mathbf{S}$ can be an arbitrary subset of $\{\mathbf{T},\mathbf{F}\}$. The resulting posteriors of parents of leaf nodes in LT can then be used for FAU intensity estimation in (7).

The bottom-up/top-down inference on LT efficiently computes the marginal posteriors $p(\mathbf{h}_l|\mathbf{S})$ in two passes – bottom-up and top-down, as illustrated in Fig. 1. In particular, for every $\mathbf{h}_l$, the algorithm defines the set of *inside* variables $\mathbf{x}_{in(l)} = \{\mathbf{x}_m : \mathbf{x}_m \in \mathbf{S}$ is descendant of $\mathbf{h}_l\}$, and the set of *outside* variables $\mathbf{x}_{out(l)} = \{\mathbf{x}_m : \mathbf{x}_m \in \mathbf{S}$ is not descendant of $\mathbf{h}_l\}$, and their distributions

$$\boldsymbol{\beta}_l = p(\mathbf{x}_{in(l)}|\mathbf{h}_l), \quad \boldsymbol{\alpha}_l = p(\mathbf{h}_l|\mathbf{x}_{out(l)}). \tag{8}$$

From (8), it is straightforward to derive that for all $\mathbf{h}_l \in \mathbf{H}$:

$$p(\mathbf{h}_l|\mathbf{S}) = \tfrac{\boldsymbol{\beta}_l\boldsymbol{\alpha}_l}{\sum_{\mathbf{h}_l}\boldsymbol{\beta}_l\boldsymbol{\alpha}_l}. \tag{9}$$

Bottom-up. The algorithm first computes the likelihoods $\boldsymbol{\beta}_l$ in the bottom-up pass starting from the leaves as

$$\boldsymbol{\beta}_l = \textstyle\prod_c (\sum_{\mathbf{h}_c} \boldsymbol{\beta}_c p(\mathbf{h}_c|\mathbf{h}_l)), \tag{10}$$

where $\{\mathbf{h}_c\}$ are children of $\mathbf{h}_l$. Note: If some $\mathbf{x}_m$ are unobserved, i.e. if $\mathbf{S}$ is a strict subset of $\{\mathbf{T},\mathbf{F}\}$, then the unobserved $\boldsymbol{\beta}_m$ are uniform.

Top-down. Then, the algorithm computes the distributions $\boldsymbol{\alpha}_l$ starting from the root as

$$\boldsymbol{\alpha}_l = \textstyle\sum_{\mathbf{h}_{P(l)}} p(\mathbf{h}_l|\mathbf{h}_{P(l)})\boldsymbol{\alpha}_{P(l)} \prod_s \left(\sum_{\mathbf{h}_s} \boldsymbol{\beta}_s p(\mathbf{h}_s|\mathbf{h}_{P(l)})\right), \tag{11}$$

where $\{\mathbf{h}_s : \mathbf{h}_{P(l)}{=}\mathbf{h}_{P(s)}, \mathbf{h}_s{\neq}\mathbf{h}_l\}$ are the siblings of $\mathbf{h}_l$.

In summary, for FAU intensity estimation, we first run the upward pass (10) and then the downward pass (11) to compute the distributions of inside and outside variables, $\boldsymbol{\beta}_l$ and $\boldsymbol{\alpha}_l$, for all hidden variables $\mathbf{h}_l \in \mathbf{H}$, and then estimate the specific marginal posterior $p(\mathbf{h}_{P(m)}|\mathbf{F})$ as in (9) required for estimating the FAU intensity $\hat{\mathbf{x}}_m$ as in (7).

As explained in the sequel, we also use the bottom-up/top-down inference algorithm as an integral part of learning model parameters. For this learning, we will be required to compute both marginal posteriors $p(\mathbf{h}_l|\mathbf{X})$ and pairwise posterior marginals $p(\mathbf{h}_l,\mathbf{h}_{P(l)}|\mathbf{X})$. Fortunately, due to the tree structure of our model, they can be computed exactly as in (9), and as

$$p(\mathbf{h}_l,\mathbf{h}_{P(l)}|\mathbf{X}) \sim \boldsymbol{\beta}_l p(\mathbf{h}_l|\mathbf{h}_{P(l)}) \prod_s (\sum_{\mathbf{h}_s} \boldsymbol{\beta}_s p(\mathbf{h}_s|\mathbf{h}_{P(l)}))\boldsymbol{\alpha}_{P(l)}, \tag{12}$$

where $\{\mathbf{h}_s\}$ are the siblings of $\mathbf{h}_l$.

4. Learning LT

Given a set of training data $\{\mathbf{X}^{(1)},...,\mathbf{X}^{(N)}\}$, we learn LT parameters and LT structure by maximizing $\mathcal{L}$, given by

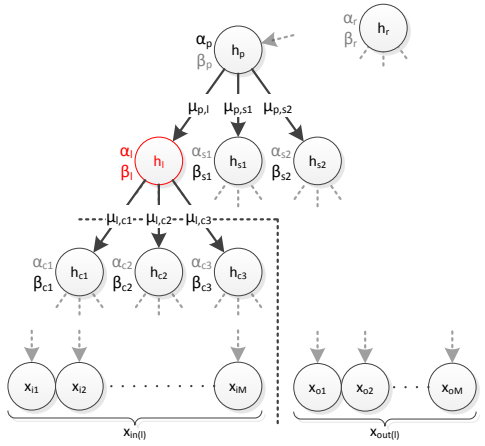

Figure 1. The inside-outside algorithm for computing the marginal posterior of node $\mathbf{h}_l$ (red) with parent $\mathbf{h}_p$, children $\{\mathbf{h}_{c1}, \mathbf{h}_{c2}, \mathbf{h}_{c3}\}$ and siblings $\{\mathbf{h}_{s1}, \mathbf{h}_{s2}\}$. $\mathbf{x}_{in(l)}$ and $\mathbf{x}_{out(l)}$ are two complementary sets of leaf nodes, where $\mathbf{x}_{in(l)}$ consists of descendants of $\mathbf{h}_l$.

(6). This maximization is conducted iteratively by alternating two steps. First, for a given current estimate of LT structure, we compute the updates of model parameters. Second, for a given current estimate of LT parameters, we conduct graph-edits for revising the LT structure. The two steps are iterated until $\mathcal{L}$ stops increasing, or the maximum number of iterations is reached. In the following, we first describe our parameter learning, and then specify our structure learning.

4.1. Learning LT Parameters

During learning of model parameters, $\{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$, defined in Sec. 2, we assume that the LT structure is given. Maximizing $\mathcal{L}$ does not lend itself to a closed-form solution, because the sum over $\mathbf{H}$ appears inside the logarithm in (6). Therefore, we resort to an EM algorithm, which iteratively estimates the joint posterior $q^{(n)} = p(\mathbf{h}_1, ..., \mathbf{h}_L|\mathbf{X}^{(n)})$, and uses $q^{(n)}$ to update the model parameters by maximizing expected log-likelihood with respect to $q^{(n)}$ as

$$(\boldsymbol{\mu}, \boldsymbol{\Sigma})^{\text{new}} = \arg\max_{(\boldsymbol{\mu}, \boldsymbol{\Sigma})} \sum_{n=1}^{N} \mathbb{E}_{q^{(n)}}[\ln p(\mathbf{X}^{(n)}, \mathbf{H})]. \quad (13)$$

Following the standard steps of finding a derivative of the expectation term in (13) with respect to each model parameter gives the well-known update equations for the Gaussian and categorial distributions of LT, presented in great detail, e.g., in [14, 3]. For completeness, we provide these model-parameter update formulas in the supplemental material.

Importantly, the update equations of parameters associated with the hidden nodes $\mathbf{h}_l$ can be expressed in terms of pairwise posterior marginals $p(\mathbf{h}_l, \mathbf{h}_{P(l)}|\mathbf{X}^{(n)})$. Also, in case of the root $\mathbf{h}_r$ and leaf nodes $\mathbf{x}_m$, the parameter update equations are expressed in terms of posterior marginals $p(\mathbf{h}_r|\mathbf{X}^{(n)})$ and $p(\mathbf{h}_{P(m)}|\mathbf{X}^{(n)})$, respectively. These posteriors can be computed exactly in (9) and (12) using the bottom-up/top-down inference algorithm.

4.2. Learning LT Structure

Given an estimate of LT parameters, our goal is to find an optimal tree structure that would maximize $\mathcal{L}$, given by (6). Finding an optimal tree in the space of all possible trees is intractable. Therefore, we specify a heuristic algorithm for structure learning. A common approach is to start from a trivial initial tree which has no connections between nodes and no hidden nodes. From there, the tree is successively altered according to an optimization criterion, e.g., using Mutual Information (MI) [8] or information distance [2], until convergence. Rather than adopting a new information-theoretic criterion for structure learning, we use the very same log-likelihood $\mathcal{L}$ for learning the tree as when learning model parameters. Our unified framework of parameter and structure learning allows us to derive an efficient algorithm for revising the tree so as to maximize log-likelihood gains.

Another common issue in structure learning is regularization of model complexity, typically addressed by using the Bayesian information criterion (BIC) [8]. As one of our contributions, we regularize our tree learning by favoring those structure changes that: 1) Minimally increase model complexity, while at the same time 2) Maximally increase the gain in the conditional likelihood of all descendant nodes under the introduced structural change. The latter regularization condition is motivated by the generative properties of our model: if we add a new child to a node, then we require that the conditional likelihoods of all its siblings be improved – not only the overall joint likelihood. This effectively means that the newly added child needs to contribute information to all its siblings.

Algorithm. Our structure learning iteratively revises candidate trees, starting from the initial forest of trivial trees wherein all $\mathbf{x}_m \in \mathbf{X}$ are independent, and paired with the corresponding hidden root, $\mathbf{h}_r$. In this initial forest, the joint log-likelihood of $\mathbf{X}$ is equal to the sum of the tree specific log-likelihoods. Our structure learning then proceeds by introducing either a new edge in the tree, or a new hidden node and appropriately connecting it to the existing ones. In particular, we consider two types of graph-edit operations:

1. Add new edge between two existing nodes $\mathbf{h}_{l'}$ and $\mathbf{h}_l$;
2. Add new parent $\mathbf{h}_{l'}$ to existing nodes $\mathbf{h}_{l_1}, \mathbf{h}_{l_2}$.

The graph-edit operations (1) and (2) yield changes in log-likelihood, Δ. Our goal is to identify the operation that produces the highest Δ and simultaneously meets the regularization constraints. One constraint is to maintain the tree structure. Another is the regularization constraint that requires, for all siblings $\{\mathbf{h}_s\}$ of the newly added node $\mathbf{h}_{l'}$, that the difference in the conditional likelihood $p(\mathbf{x}_{in(s)}|\mathbf{x}_{out(s)})$ must be greater than a threshold. The structure learning terminates if there are no graph revisions to perform, i.e., when the tree becomes rooted at a sin-

gle root, or all possible graph-edits would lead to a log-likelihood decrease.

Efficiency. In the above algorithm, Δ has to be calculated for all possible pairs $(\mathbf{h}_l, \mathbf{h}_{l'})$, which is quadratic in the number of roots. We specify two mechanisms to achieve efficiency. The first mechanism concerns our observation that adding new nodes by graph-edit operation (2) will increase model complexity more than operation (1), since (1) adds just an edge, whereas (2) adds a node and two edges. Therefore, we specify a heuristic procedure to first evaluate the Δ's of all possible operations of type (1), and start considering (2) only if none of operations of type (1) meet the above algorithm's criteria and constraints. The second mechanism concerns our efficient evaluation of Δ. Specifically, after the structural change, we perform a single M-step to compute only the model parameters for the newly added connection between $\mathbf{h}_l$ and $\mathbf{h}_{l'}$, while using the joint posterior q from the previous E-step, before the structural change. It is straightforward to show that this approximate procedure is guaranteed to increase the log-likelihood in the M-step [14, 3].

5. Results

In order to evaluate the LT model for FAU recognition, we design the experiments to contain local targets and local features, so that out LT model can discover the hidden structure that governs the dependencies of the input, which in this application leads to a joint generative model of the facial points and the FAUs. In the following sections, we describe the datasets used (Sec. 5.1), how the datasets were used for evaluation (Sec. 5.2), the models that we compare to (Sec. 5.3), which metrics we use for evaluation (Sec. 5.4) followed by quantitative (Sec. 5.5) and qualitative (Sec. 5.6) results.

5.1. Data

In this study, we focus on the DISFA [18] and ShoulderPain [16] datasets, since they both provide per-frame FAU intensity labels for spontaneous facial expressions.

The DISFA dataset [18] contains spontaneous facial expressions of young adults while watching an emotion eliciting video. The face of 27 subjects was recorded with a total number of 130754 frames. Each of the 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 (see [18] for the FAU distribution). Additionally, the database provides 66 active-appearance-model (AAM) tracked facial landmark points.

The ShoulderPain data [16] consists of videos showing faces of patients suffering from shoulder pain while moving their arms. 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, the intensities of pain related FAU's 4, 6, 7, 9, 10, 12, 20, 25, 26, 27 and 43 are included with the database. As above, all FAUs are annotated on a 0-5 discrete intensity scale, except for FAU 43, which is binary. We exclude FAU 27, since it is present for 18 frames only.

In order to obtain local point features (PTS) of the face, we follow the same procedure for both DISFA and ShoulderPain: the 66 tracked facial landmarks are aligned by Procrustes analysis to the mean shape, which removes translations and in-plane rotations of the face. Then each of the x and y landmark coordinates are normalized by subtracting the mean and dividing by the standard deviation. The coordinates are stacked together into the final 132 dimensional feature vector.

Additionally we extract appearance features from the DISFA data by first normalizing the face image using a piece-wise affine warp to the mean shape. Then we divide the face into 6x6 non-overlapping patches and extract local binary pattern histograms (LBP) [19] with 59 bins from each patch, resulting into into a 36x59 dimensional feature vector.

Each of the shape feature dimensions is continuous and thus modeled in the LT with a Gaussian node. The FAU targets are discrete and thus modeled with a Categorical node, where each category corresponds to one FAU intensity level. The LBP features consist of histograms with 59 bins and therefore we model each of them with a Categorical node having 59 states. For prediction, we use the expected value of the FAU intensity, given the corresponding posterior node distribution. This means the prediction are continuous, but restricted to the interval 0-5.

5.2. Evaluation Setting

We evaluate the model in a subject-independent setting, i.e. different subjects were used for training than for testing. The data is grouped into cross-validation folds with no more than 3 subjects per fold, which leads to 9 folds for DISFA and 8 folds for ShoulderPain.

The LT model supports the prediction of multiple FAU targets at the same time, but in order to determine if multiple targets improve the performance, we evaluate our model for different settings: (1) LT-all, which includes all FAUs for training; (2) LT-sep which trains a separate model for each FAU; and (3) LT-single, which is limited to a single hidden variable and trained separately per FAU as LT-sep.

Additionally to the evaluation on clean data, we create random noise to corrupt the test features. The noise is created with different severity levels: 50% noise features means that for every testing instance, we randomly select 50% of the feature dimensions and replace them with a randomly sampled value from a Gaussian distribution that has the same mean and variance as the overall training dataset.

The noise is only influencing the test data, i.e. the models are trained on clean data.

5.3. Baseline Methods

We compare our method to Support Vector Classification (SVC), Support Vector Regression (SVR) (both using LIBSVM [1]) and Binary Latent Trees (BLT) [8].

SVC has been used for the baselines of DISFA [18] and ShoulderPain [16], by treating each of the intensity levels as a separate class and applying the one-vs-one approach. SVR is similar, but it treats all target intensities on a continuous scale, rather than separate categories. We apply the Gaussian kernel to SVC and SVR and optimize all hyper-parameters, by a grid search. SVC and SVR support only a single target, therefore we train a separate model per FAU.

BLT has not been used in a supervised context, but the inference step can infer the unobserved targets given observed features. Furthermore, BLT allows only categorical nodes, therefore we first apply k-means clustering with $K = 10$ to each of the continuous feature dimensions and then use the assigned cluster as categorical feature.

5.4. Metrics

The goal is FAU *intensity* prediction and therefore we measure the performance by several continuous metrics: the Pearson correlation coefficient (CORR), the mean squared error (MSE) and the Intra-Class Correlation Coefficient ICC(3,1) [24]. The former two are common within the machine learning community and measure relative (CORR) and absolute (MSE) differences between target and prediction, while the latter ICC comes from behavioral sciences and measures agreement between raters. Additionally we compare our LT-all model to all other methods by the pairwise Student's t-test with a p-value of 0.05 and mark all significantly different results with '*'.

5.5. Quantitative Results

First, the DISFA results for different feature combinations are shown, followed by detailed shape feature results on the DISFA and ShoulderPain databases.

DISFA Data. Tab. 1 shows the average CORR over all FAUs on the DISFA data for different feature combinations. The models have been evaluated using point (PTS), appearance (LBP) and the combination of both (PTS+LBP). The reported results for LBP are consistently lower than the ones for PTS. This can be explained with the nature of LBPs: since they aggregate information within a histogram, the locality of the data is lost and thus it is difficult for our model to learn local distributions, represented by branches of the tree. Combining LBP and PTS gives no improvement over PTS alone, therefore all following results are shown for PTS features only. We were not able to obtain LBP and PTS+LBP results for BLT, since the algorithm did not terminate after running for 72 hours due to the high dimensionality of the data.

Feature	PTS	LBP	PTS+LBP
LT-all	0.43	0.14	0.43
LT-sep	0.41	0.10	0.40
LT-single	0.33	0.12	0.33
SVC	0.23	0.21	0.28
SVR	0.43	0.34	0.43

Table 1. Average results over all FAU targets measured by the correlation coefficient (CORR) on DISFA data for different features. We compare facial landmark points (PTS) with local binary pattern (LBP) features and also show the combined results (PTS+LBP).

Tab. 2 shows the results on the DISFA dataset for PTS features. LT-all is on average the best for all measures and the results for LT-sep are slightly lower, which shows that it is beneficial to learn all FAUs together. SVC is significantly inferior than LT-all in almost all cases. This is probably due the discriminative nature of the SVC approach, which is better suited for binary classification. LT-all is significantly better than SVR regarding most measures for the FAUs 5, 12, and 25 and LT-all is significantly worse than SVR regarding the most measures for the FAUs 9 and 15. This can be explained by the fact that FAUs 5, 12, and 25 are pronounced in the localized model, since they elicit large local variation in the points which is more difficult to be captured by the SVR that uses a kernel over all dimensions. In contrast to that, FAUs 9 and 15 are barely recognizable with facial landmarks since they induce very little movement over a larger set of points, which is not captured in our generative model. BLT is significantly inferior to LT-all in most cases, because it creates a tree with more hidden nodes, and thus the features and targets are farther apart in the tree, which leads to lower dependence. Often BLT will not connect all data input nodes, which leaves a forest with mutual independent sets of variables. In contrast to that, the LT nodes are fully connected and each hidden node has on average 3.39 children and cardinality K=10 (K is optimized by a grid search). A typical tree has a relatively deep structure.

The last two columns of Tab. 2 show the average results for 10% (a[10]) and 20% (a[20]) added noise. Although the CORR of LT-all and SVR is on par for 0% noise, the LT-model does better as the noise increases. This effect is even more pronounced in Fig. 3, which shows the results for the average CORR and selected FAUs as the noise level varies. The result at 0% noise is the same value as in Tab. 2 and the performance deteriorates as the noise increases. The performance drop of our LT model is slower than the other models, and it is even possible to beat other models as the noise increases, see FAU17: at about 50% noise, out perfor-

	FAU	1	2	4	5	6	9	12	15	17	20	25	26	a[0]	a[10]	a[20]
CORR	LT-all	.41	**.44**	.50	.29	.55	.32	.76	.11	.31	.16	**.82**	.49	**.43**	**.40**	**.36**
	LT-sep	.41	**.44**	.47	**.34***	.55	.27	**.77**	.09	.18	.10	**.82**	.47	.41		
	LT-single	.30*	.29*	.27*	.12*	**.56**	.21*	.74	.09	.16	.12	.76*	.39*	.33		
	SVC	.19*	.19*	.33*	.01*	.10*	.12*	.60*	.02*	.14*	.04	.71*	.33*	.23*	.20*	.17*
	SVR	**.42**	**.44**	**.53**	.15*	.47	**.43***	.70*	**.21***	**.32**	**.21**	.76*	**.51**	**.43**	.39	.34
	BLT	.04*	.05*	.20*	.00*	.55	.18	.73	.01*	.04*	.02	**.82**	.26*	.24*	.23*	.22*
MSE	LT-all	.44	.39	.96	**.07**	.41	.31	.40	.17	.33	.16	.61	.46	**.39**	**.42**	**.46**
	LT-sep	**.41**	.37	1.00	**.07**	**.40**	.31	**.39**	.16	.33	**.15**	**.58**	.46	**.39**		
	LT-single	.47	.41	1.21*	**.07**	.41	.32	.44	.16	.32	**.15**	.76*	.50	.43		
	SVC	.51	.43	1.21	.08	.65*	.34	.65*	.18	.35	.16	.99*	.56*	.51*	.53*	.55*
	SVR	.42	**.35**	**.87**	**.07**	.45	**.27***	.50*	**.15***	**.29***	**.15**	.76*	**.41***	**.39**	**.42**	**.46**
	BLT	.53*	.45	1.27*	**.07**	**.40**	.31	.47	.16	.33	**.15**	.63	.56*	.45*	.45	.47
ICC	LT-all	**.32**	**.37**	.41	**.18**	**.46**	.23	**.73**	.07	**.23**	.09	**.80**	.39	**.36**	**.33**	**.31**
	LT-sep	.26*	.29*	.39	.15	.44	.18	**.73**	.06	.11	.03	**.80**	.39	.32		
	LT-single	.15*	.15*	.17*	.04*	.45	.12*	.70	.04	.07*	.06	.74*	.30*	.25*		
	SVC	.12*	.11*	.31	.00*	.09*	.08*	.58*	.01*	.11*	.02	.70*	.28*	.20*	.17*	.14*
	SVR	.28	.30*	**.44**	.09*	.36	**.29**	.62*	**.13***	**.23**	**.12**	.71*	**.42**	.33	.28*	.23*
	BLT	.03*	.03*	.12*	.00*	.45	.08*	.68*	.00*	.01*	.00	**.80**	.17*	.20*	.19*	.19*

Table 2. Results on the DISFA data for different FAU targets and PTS features. Different LT models are compared to SVC, SVR and BLT. The table shows CORR, MSE and ICC measures. The best results per FAU and per measure are marked in bold. The results that are statistically different to LT-all are marked with *. Additionally we show the average performance over all FAUs (a[0]), as well as the average performance for 10% and 20% added noise (a[10] and a[20]).

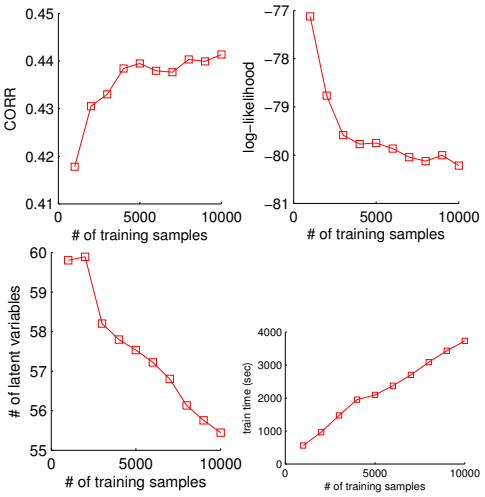

Figure 2. Results on the DISFA data while training our LT-all model on different number of training samples. Shown is the CORR performance, the log-likelihood and the number of latent variables of the trained model, as well as the training time.

mance is better than SVR, although SVR has the better performance on clean data. This is due to our generative model, which is able to ignore noisy data that is not consistent with the clean dimensions. This effect is also pronounced with the BLT model: BLT has the same average performance on clean data as SVM. However, with increasing noise, BLT performs clearly better than SVM.

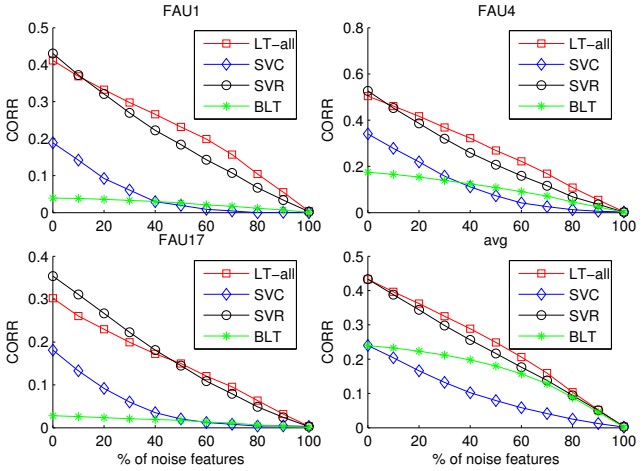

Figure 3. Results on the DISFA data for different FAU targets. The LT-all model is compared to SVC, SVR and BLT. Each graph shows the correlation (CORR) as the percentage of noise feature varies.

ShoulderPain Data. Tab. 3 shows the results for the ShoulderPain data for different FAU targets. The LT model does best in most of the cases, except for FAU 7 and 26. The appearance of these FAUs is only barely present within the points and thus a generative model will assume theses changes to be noise, whereas discriminative models can learn them. The recognition results are in general lower than for the DISFA data, which is due to less frequent FAU occurrences in the data and larger head movements.

	FAU	4	6	7	9	10	12	20	25	26	43	avg
CORR	LT-all	.03	**.60**	.11	**.10**	**.15**	**.60**	**.09**	**.18**	.01	**.44**	**.23**
	SVC	.04	.45	.25	.02	.06	.45	.00	.13	.07	.30	.18
	SVR	**.05**	.48	**.26**	.09	.10	.44	.03	.17	**.10**	**.44**	.22
	BLT	.03	.55	.06	.05	.05	.55	.00	.07	.06	.21	.16
MSE	LT-all	.51	**1.06**	**1.19**	**.27**	**.28**	**1.12**	**.19**	.72	.50	**.14**	**.60**
	SVC	.76	1.74*	1.59	.48	.32	1.54	.36	1.24	.56	.17	.88
	SVR	.65	1.44	1.40	.40	.36	1.35	.30	.76	.76	.15	.76
	BLT	**.48**	1.15	1.29	**.27**	.31	1.17	**.19**	**.66**	**.41**	.18	.61

Table 3. Results on the ShoulderPain data for different FAU targets. The L-all model is compared to SVC, SVR and BLT. The table shows the correlation (CORR) and mean squared error (MSE). The best results per FAU and per measure are marked in bold. The results that are statistically different to LT-all are marked with *.

Fig. 4 shows the CORR results on ShoulderPain for varying feature noise. Again, the LT model can handle the noise well and stays above the competing methods in the most cases. The advantage is less pronounced than in the DISFA data, which can be due to the less descriptive clean data, i.e. the facial movements are less pronounced than in DISFA.

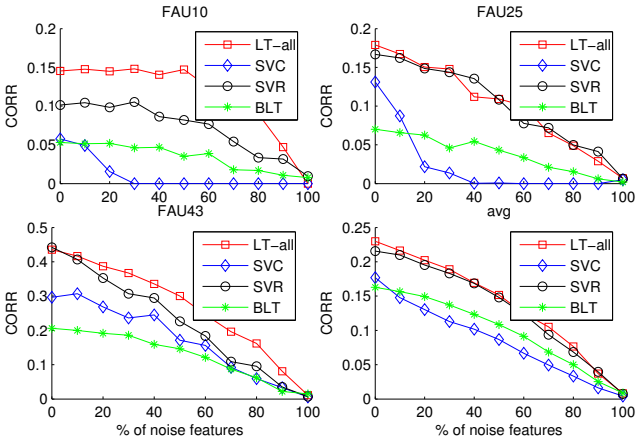

Figure 4. Results on the ShoulderPain data for different FAU targets. The LT-all model is compared to SVC, SVR and BLT. Each graph shows the correlation (CORR) as the percentage of noise feature varies.

Fig. 2 shows the CORR, likelihood, number of latent variables and the training time for our LT-all model as the number of training samples varies. The CORR starts to level out after 4000 samples. The likelihood and the number of latent variables both decrease when the training samples increase. This shows that the model overfits with few examples, i.e. it has a high likelihood and many latent variables. However, with increasing number of training samples, the overfitting vanishes and thus the likelihood decreases. The training time shows a clear linear trend. Furthermore, the LT inference time for recognizing all 12 FAUs in 14,535 testing samples is 17.9 sec, where SVR takes 106.2 sec for the same task.

5.6. Qualitative Results

Fig. 5 shows the face model generated by LT on the DISFA data. Given that the LT model gets all FAUs zero as input (i.e. the neutral face), we plot the inferred mean of the model output distribution in black. Then we overlay in red the changed landmark means if the models gets the maximum FAU intensity as input. We can clearly see the correspondence between the mean differences and the facial regions influenced by the related FAU, e.g. FAUs 1 mainly influences the eyebrow landmarks while FAU 12 mainly influences the landmarks around the mouth. The landmarks for FAU15 have almost no difference to the neutral face, which shows that the model was not able to learn the subtle movements. This explains the low results in Tab. 2 for FAU15.

5.7. Comparison with prior work

Several previous publications have already addressed the FAU intensity estimation problem within the DISFA and ShoulderPain databases.

Figure 5. Landmark locations generated from an LT model trained on the DISFA data. Shown are the landmark means for the generated neutral face (in black) and the changed mean locations for the face with activated FAU (in red).

A dynamic ordinal regression framework is developed in [21], which reaches an average ICC of 0.58 on the DISFA and 0.62 on the ShoulderPain data. However, the model is evaluated on pre-segmented video sequences that both start and end with a neutral face Thus the results cannot directly compete within our setting, which does per-frame FAU recognition without prior knowledge.

The work of [22] estimates the FAU intensities by SVR and as a second step models the dependencies between FAUs by a Markov random field. This work also evaluates the performance of a tree structure, reaching an average CORR result of 0.34 for the FAUs 1, 2, 4, 5, 6 and 9, where our average is 0.42.

Furthermore, [18, 15] learn a manifold of facial features and use SVC [18] or DBN [15] on the manifold to classify different FAU intensities, reaching an average ICC of 0.77. However the comparison to our method is not fair, since the supervised FAU specific manifold learning includes the FAU targets of the test subjects and thus the methods are not subject-independent.

Different features and relevance vector regression is used within [12] on the ShoulderPain data, reaching an average CORR of 0.36 by fusing different appearance features. However the points alone, which is equivalent to our setting, reach only an average CORR of 0.17.

6. Conclusion

We formulated a novel latent tree (LT) model for FAU intensity estimation in videos based on locations of facial landmark points in each frame. For learning LT structure, we specified an efficient algorithm that iteratively maximizes log-likelihood of training data while maintaining model complexity low. In our comparison with discriminative approaches on the benchmark DISFA and ShoulderPain datasets, LT produced superior results, especially in realistic settings of noisy detections of facial landmark points. Probabilistic sampling from LT generated meaningful facial expressions, demonstrating good generalization capabilities of LT and effectiveness of our structure learning algorithm in capturing higher-order dependencies among the high-dimensional input features and target FAU intensities.

Acknowledgements

This work has been funded by the European Community Horizon 2020 [H2020/2014-2020] under grant agreement no. 645094 (SEWA). The work of Sebastian Kaltwang was previously funded in part by the EPSRC Emotion & Pain Project [EP/H016988/1]. The work of Sinisa Todorovic has been funded by the grant NSF RI 1302700.

References

[1] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, 2011.

[2] M. J. Choi, V. Y. F. Tan, A. Anandkumar, and A. S. Willsky. Learning Latent Tree Graphical Models. *J. Mach. Learn. Res.*, 12:1771–1812, July 2011.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Stat. Soc. Ser. B*, 39(1):1–38, 1977.

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

[5] 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.

[6] N. Friedman. The Bayesian Structural EM Algorithm. In *Proc. 14th Conf. Uncertain. Artif. Intell.*, UAI'98, pages 129–138, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[7] S. D. Gunnery, J. A. Hall, and M. A. Ruben. The Deliberate Duchenne Smile: Individual Differences in Expressive Control. *J. Nonverbal Behav.*, 37(1):29–41, 2013.

[8] S. Harmeling and C. K. I. Williams. Greedy Learning of Binary Latent Trees. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(6):1087–1097, June 2011.

[9] 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.*, 69(2):280–288, 1995.

[10] U. Hess, S. Blairy, and R. E. Kleck. The intensity of emotional facial expressions and decoding accuracy. *J. Nonverbal Behav.*, 21(4):241–257, 1997.

[11] B. Jiang, B. Martinez, M. Valstar, and M. Pantic. Automatic Analysis of Facial Actions: A Survey. *Int. J. Comput. Vis.*

[12] S. Kaltwang, O. Rudovic, and M. Pantic. Continuous Pain Intensity Estimation from Facial Expressions. In G. Bebis, R. Boyle, B. Parvin, D. Koracin, C. Fowlkes, S. Wang, M.-H. Choi, S. Mantler, J. Schulze, D. Acevedo, K. Mueller, and M. Papka, editors, *Adv. Vis. Comput.*, volume 7432 of *Lecture Notes in Computer Science*, pages 368–377, Heidelberg, 2012. Springer.

[13] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[14] P. F. Lazarsfeld and N. W. Henry. *Latent Structure Analysis*. Houghton Mifflin, 1968.

[15] Y. Li, S. M. Mavadati, M. H. Mahoor, and Q. Ji. A unified probabilistic framework for measuring the intensity of spontaneous facial action units. In *IEEE Int.Conf. Autom. Face Gesture Recognit.*, pages 1–7, Apr. 2013.

[16] 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.*, pages 57–64. IEEE, 2011.

[17] D. Matsumoto and B. Willingham. Spontaneous facial expressions of emotion of congenitally and noncongenitally blind individuals. *J. Personal. Soc. Psychol.*, 96(1):1–10, 2009.

[18] S. Mavadati, M. Mahoor, K. Bartlett, P. Trinh, and J. F. Cohn. DISFA: A Spontaneous Facial Action Intensity Database. *IEEE Trans. Affect. Comput.*, 4(2):151–160, 2013.

[19] 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.*, 24(7):971–987, 2002.

[20] A. Ortony and T. J. Turner. What's basic about basic emotions? *Psychol. Rev.*, 97(3):315–331, 1990.

[21] 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.*, 37(5):944–958, 2015.

[22] G. Sandbach, S. Zafeiriou, and M. Pantic. Markov Random Field Structures for Facial Action Unit Intensity Estimation. In *IEEE Int. Conf. Comput. Vis. Work.*, pages 738—-745, Dec. 2013.

[23] K. R. Scherer and H. Ellgring. Are facial expressions of emotion produced by categorical affect programs or dynamically driven by appraisal? *Emotion*, 7(1):113–130, 2007.

[24] P. E. Shrout and J. L. Fleiss. Intraclass correlations: uses in assessing rater reliability. *Psychol. Bull.*, 86(2):420–428, 1979.

[25] Y. Tong, W. Liao, and Q. Ji. Facial action unit recognition by exploiting their dynamic and semantic relationships. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(10):1683–1699, 2007.

[26] A. C. d. C. Williams. Facial expression of pain: An evolutionary account. *Behav. Brain Sci.*, 25(04):439–455, 2002.

[27] Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang. A Survey of Affect Recognition Methods: Audio, Visual, and Spontaneous Expressions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(1):39–58, 2009.