Continuous Conditional Neural Fields for Structured Regression

Tadas Baltrušaitis[1], Peter Robinson[1], and Louis-Philippe Morency[2]

[1] Computer Laboratory, University of Cambridge, UK
{tadas.baltrusaitis,peter.robinson}@cl.cam.ac.uk
[2] Institute for Creative Technologies, University of Southern California, CA
morency@ict.usc.edu

Abstract. An increasing number of computer vision and pattern recognition problems require structured regression techniques. Problems like human pose estimation, unsegmented action recognition, emotion prediction and facial landmark detection have temporal or spatial output dependencies that regular regression techniques do not capture. In this paper we present continuous conditional neural fields (CCNF) – a novel structured regression model that can learn non-linear input-output dependencies, and model temporal and spatial output relationships of varying length sequences. We propose two instances of our CCNF framework: Chain-CCNF for time series modelling, and Grid-CCNF for spatial relationship modelling. We evaluate our model on five public datasets spanning three different regression problems: facial landmark detection in the wild, emotion prediction in music and facial action unit recognition. Our CCNF model demonstrates state-of-the-art performance on all of the datasets used.

Keywords: Structured regression, Landmark detection, Face tracking.

1 Introduction

As an extension to the conventional regression problem, structured regression algorithms are designed to take advantage of the relationships between output variables. A number of computer vision problems such as human pose estimation and unsegmented action recognition exhibit temporal output structure. Structured regression is also desirable when modelling 2D spatial relationships for landmark detection problems. Another important aspect of many prediction problems is the need for modelling variable length sequences or variable size 2D grids. This is both because we might have varying length video sequences as training data; and because we would like our approaches to generalise to sequences of arbitrary length.

In this paper we present the *continuous conditional neural field* (CCNF) model that can perform structured regression. The key features of CCNF are: (1) a structured regression model; (2) captures non-linearities between input and output; (3) can easily define temporal and spatial relationships (long and short

© Springer International Publishing Switzerland 2014

distance); (4) ability to model arbitrary and different length sequences; (5) simple and efficient single pass inference.

We propose two instances of our CCNF framework: Chain-CCNF for time series modelling, and Grid-CCNF for spatial relationship modelling. We evaluate CCNF on five public datasets spanning three very different and challenging regression problems: facial landmark detection in the wild, emotion prediction in music and facial action unit recognition.[1] This work is a generalisation of our work on Grid-CCNF [1].

First we present a brief overview of structured prediction and regression techniques (Section 1.1). This is followed by the description of our CCNF model (Section 2). The two instances of the model are evaluated in Sections 3 and 4.

1.1 Prior Work

The most often used techniques for regression problems in computer vision and pattern recognition communities are linear and logistic regression, support vector regression (SVR), neural networks, and relevance vector machines (RVM) [2]. These approaches are designed to model input-output dependencies disregarding the output-output structure. SVR models have been used for modelling alignment probabilities of facial landmarks [3,4], inferring continuous emotions from music [5] and human behaviour [6], and the recognition of facial action units in videos [7]. RVM is another popular approach and has been used for emotion prediction from facial expressions [8] and speech [9].

The greatest amount of work on structured prediction concentrates on discrete outputs (classification). Popular approaches include conditional random fields (CRF) [10], Hidden Markov Models [2] structural SVM [11] and Markov Random Field models [12]. Another example of a structured classification method that is suitable for temporal modelling is the Conditional Neural Fields Model [13], which augments CRF with the ability to model non-linear relationships. It is possible to convert regression problems to classification ones through quantisation. However, this can lead to loss of information and of relationships between neighbouring classes, moreover, the number of classes to be used is often unclear.

There has been some recent work exploring structured regression as well. One such example is the Output-Associative Relevance Vector Machine (OA-RVM), that has been used for emotion prediction from facial expressions [8]. Other examples include SVR models for structured output [14] and the twinned Gaussian process model that exploits the output dependencies of multi-variate output [15]. Both of these approaches have been used for human pose estimation. However, few of the above mentioned approaches are suitable for modelling arbitrary length sequences, such as varying length time-series, directly.

Another structured regression approach is the continuous conditional random fields (CCRF) [16] model, that extends the CRF model to the continuous case. CCRF model relies on initial prediction from an unstructured regression model

[1] Code available at `https://github.com/TadasBaltrusaitis/CCNF`

such as SVR [17], making the training procedure more complex. Our approach does not suffer from this problem as all of its parameters are optimised jointly.

2 Continuous Conditional Neural Fields

In this section we introduce and define our CCNF model for structured regression. The model definition is presented in Section 2.1, followed by description of learning and inference in Section 2.2. Section 2.3 presents two specific instantiations of our model.

2.1 Model Definition

In our discussion we adopt the following notation: $\boldsymbol{x} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ is a set of observed input variables ($\mathbf{x}_i \in \mathcal{R}^m$), $\mathbf{y} = \{y_1, y_2, \dots, y_n\}$ is a set of output variables ($y_i \in \mathcal{R}$) that we wish to predict, n is the size of the set (length of sequence or the number of pixels we are interested in), and m is the dimensionality of input vector.

CCNF is an undirected graphical model that can learn the conditional probability of a continuous valued vector $\mathbf{y}$ depending on continuous $\boldsymbol{x}$. A graphical illustration of our two model instances – Chain-CCNF and Grid-CCNF, can be seen in Figure 1. We define our CCNF model as:

$$P(\mathbf{y}|\boldsymbol{x}) = \frac{\exp(\Psi)}{\int_{-\infty}^{\infty} \exp(\Psi) d\mathbf{y}} \tag{1}$$

Our potential function is defined as:

$$\begin{array}{l} \Psi = \sum_i \sum_{k=1}^{K1} \alpha_k f_k(y_i, \boldsymbol{x}, \boldsymbol{\theta}_k) + \sum_{i,j} \sum_{k=1}^{K2} \beta_k g_k(y_i, y_j) + \\ \sum_{i,j} \sum_{k=1}^{K3} \gamma_k l_k(y_i, y_j) \end{array}, \tag{2}$$

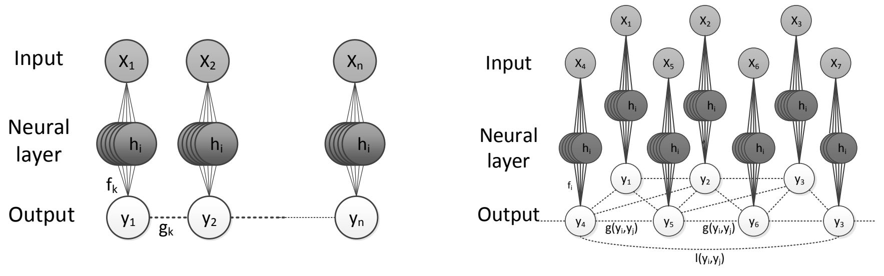
We define three types of feature functions in our model – vertex features (f_k) and two types of edge features (g_k, l_k), see Figure 1 for an illustration.

Vertex features f_k represent the mapping from the input $\mathbf{x}_i$ to output y_i through a single layer neural network and $\boldsymbol{\theta}_k$ is the weight vector for a particular neuron k. The corresponding α_k for vertex feature f_k represents the reliability of the k^{th} neuron.

$$f_k(y_i, \boldsymbol{x}, \boldsymbol{\theta}_k) = -(y_i - h(\boldsymbol{\theta}_k, \mathbf{x}_i))^2, \tag{3}$$

$$h(\boldsymbol{\theta}, \mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^T \mathbf{x}}}; \tag{4}$$

Edge features g_k represent the similarities between observations y_i and y_j, allowing the model to enforce smoothness. This is controlled by the neighbourhood measure $S^{(g_k)}$, which allows us to control where the smoothness is to be enforced (which nodes should be connected). If $S_{i,j}^{(g_k)} > 0$ there exists a similarity connection between y_i and y_j; if $S_{i,j}^{(g_k)} = 0$ they are not connected. It is



(a) Chain-CCNF (b) Grid-CCNF

Fig. 1. Two instances of our CCNF model: Chain-CCNF and Grid-CCNF. Solid lines represent vertex features (f_k), dashed lines represent edge features (g_k or l_k). The input vector $\mathbf{x}_i$ is connected to the relevant output y_i through the vertex features that combine the neural layer (Θ) and the vertex weights $\boldsymbol{\alpha}$. The outputs are further connected with edge features g_k (similarity) or l_k (sparsity). Only direct links from $\mathbf{x}_i$ to y_i are presented here, but extensions are straightforward.

important to note that CCNF supports connections between any of the nodes, allowing us to create long range dependencies and still retain tractable training and inference.

$$g_k(y_i, y_j) = -\frac{1}{2} S_{i,j}^{(g_k)} (y_i - y_j)^2; \tag{5}$$

Edge features l_k represent the sparsity (or inhibition) constraint between observations y_i and y_j. For example the model is penalised if both y_i and y_j are high, but is not penalised if both of them are zero (note that we are constrained to certain types of edge and vertex features in order to keep model learning and inference tractable). This is controlled by the neighbourhood measure $S^{(l_k)}$ that allows us to define regions where sparsity/inhibition should be enforced. This feature is particularly useful if we want to model output as a unimodal probability density (which is the case for landmark detection).

$$l_k(y_i, y_j) = -\frac{1}{2} S_{i,j}^{(l_k)} (y_i + y_j)^2. \tag{6}$$

The feature functions are parametrised by parameters that need to be learned: $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \ldots \alpha_{K1}\}$, $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots \boldsymbol{\theta}_{K1}\}$, $\boldsymbol{\beta} = \{\beta_1, \beta_2, \ldots \beta_{K2}\}$, and $\boldsymbol{\gamma} = \{\gamma_1, \gamma_2, \ldots \gamma_{K3}\}$. The number of edge feature functions (K2, and K3) is dependent on the CCNF instance (see Section 2.3) and the number of vertex feature functions K1 depends on the nature of the regression problem and can be determined during model validation.

2.2 Learning

In this section we describe how to estimate the parameters $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\Theta}\}$. It is important to note that all of the parameters are optimised jointly. We learn the temporal/spatial structure alongside the mapping from features to the prediction, which is not the case in a model like CCRF.

We are given training data $\{\mathbf{x}^{(q)}, \mathbf{y}^{(q)}\}_{q=1}^{M}$ of M sets, where each $\boldsymbol{x}^{(q)} = \{\mathbf{x}_1^{(q)}, \mathbf{x}_2^{(q)}, \ldots, \mathbf{x}_n^{(q)}\}$ is a set of inputs and each $\mathbf{y}^{(q)} = \{y_1^{(q)}, y_2^{(q)}, \ldots, y_n^{(q)}\}$ is a set of real valued outputs, n can be different for each set.

In learning we want to optimise the $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$ and $\boldsymbol{\Theta}$ parameters that maximise the conditional log-likelihood of CCNF on the training sequences:

$$L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\Theta}) = \sum_{q=1}^{M} \log P(\mathbf{y}^{(q)}|\boldsymbol{x}^{(q)}) \tag{7}$$

$$(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*, \boldsymbol{\gamma}^*, \boldsymbol{\Theta}^*) = \underset{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\Theta}}{\arg\max}(L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\Theta})) \tag{8}$$

Because of the careful vertex and edge feature selection, Equation 1 can be transformed into a multivariate Gaussian form (a more detailed derivation can be found in the Appendix in the supplementary material):

$$P(\mathbf{y}|\boldsymbol{x}) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} \exp(-\frac{1}{2}(\mathbf{y} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{y} - \boldsymbol{\mu})). \tag{9}$$

This is achieved by collecting the quadratic $\boldsymbol{y}$ terms in the exponent into the covariance matrix:

$$\Sigma^{-1} = 2(A + B + C) \tag{10}$$

$$A_{i,j} = \begin{cases} \sum_{k=1}^{K1} \alpha_k, \; i = j \\ 0, \; i \neq j \end{cases} \tag{11}$$

$$B_{i,j} = \begin{cases} (\sum_{k=1}^{K2} \beta_k \sum_{r=1}^{n} S_{i,r}^{(g_k)}) - (\sum_{k=1}^{K2} \beta_k S_{i,j}^{(g_k)}), \; i = j \\ -\sum_{k=1}^{K2} \beta_k S_{i,j}^{(g_k)}, \; i \neq j \end{cases} \tag{12}$$

$$C_{i,j} = \begin{cases} (\sum_{k=1}^{K2} \gamma_k \sum_{r=1}^{n} S_{i,r}^{(l_k)}) + (\sum_{k=1}^{K2} \gamma_k S_{i,j}^{(l_k)}), \; i = j \\ \sum_{k=1}^{K2} \gamma_k S_{i,j}^{(l_k)}, \; i \neq j \end{cases} \tag{13}$$

The diagonal matrix A represents the contribution of $\boldsymbol{\alpha}$ terms (vertex features) to the covariance matrix, and the symmetric B and C represent the contribution of the $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$ terms (edge features).

It is useful and convenient for inference to define a vector $\boldsymbol{d}$, that describes the linear terms in the distribution, and $\boldsymbol{\mu}$ which is the mean value of the Gaussian form of the CCNF distribution:

$$\mathbf{d} = 2\boldsymbol{\alpha}^T(1 + \exp(-\boldsymbol{\Theta}\mathbf{X}))^{-1}, \tag{14}$$

$$\boldsymbol{\mu} = \Sigma \boldsymbol{d}. \tag{15}$$

Above $\mathbf{X}$ is a matrix where the i^{th} column is $\mathbf{x}_i$, $\boldsymbol{\Theta}$ is the concatenated neural network weights, and exp is an element-wise exponential function.

Intuitively $\boldsymbol{d}$ is the contribution from the the vertex features. These are the terms that contribute directly from input features $\mathbf{x}$ towards $\mathbf{y}$. Σ on the other hand, controls the influence of the edge features on the output. Finally, $\boldsymbol{\mu}$ is the expected value of the distribution.

In order to guarantee that our partition function is integrable, we constrain $\alpha_k > 0$ and $\beta_k > 0, \gamma_k > 0$ [16]. The log-likelihood can be maximised using a gradient based method such as constrained limited memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm [18]. In order to make the optimisation more accurate and faster we the partial derivatives of the $\log P(\mathbf{y}|\boldsymbol{x})$ can be used:

$$\frac{\partial \log(P(\boldsymbol{y}|\boldsymbol{x}))}{\alpha_k} = -\boldsymbol{y}^T\boldsymbol{y} + 2\boldsymbol{y}^T H_{k,*}^T - 2H_{*,k}\boldsymbol{\mu} + \boldsymbol{\mu}^T\boldsymbol{\mu} + \mathrm{tr}(\Sigma) \tag{16}$$

$$\frac{\partial \log(P(\boldsymbol{y}|\boldsymbol{x}))}{\beta_k} = -\boldsymbol{y}^T\frac{\partial B}{\partial \beta_k}\boldsymbol{y} + \boldsymbol{\mu}^T\frac{\partial B}{\partial \beta_k}\boldsymbol{\mu} + \mathrm{tr}(\Sigma\frac{\partial B}{\partial \beta_k}), \tag{17}$$

$$\frac{\partial \log(P(\boldsymbol{y}|\boldsymbol{x}))}{\gamma_k} = -\boldsymbol{y}^T\frac{\partial C}{\partial \gamma_k}\boldsymbol{y} + \boldsymbol{\mu}^T\frac{\partial C}{\partial \gamma_k}\boldsymbol{\mu} + \mathrm{tr}(\Sigma\frac{\partial C}{\partial \gamma_k}), \tag{18}$$

$$\frac{\partial \log(P(\boldsymbol{y}|\boldsymbol{x}))}{\theta_{i,j}} = \boldsymbol{y}^T\frac{\partial \boldsymbol{b}}{\partial \theta_{i,j}} - \boldsymbol{\mu}^T\frac{\partial \boldsymbol{b}}{\partial \theta_{i,j}} \tag{19}$$

$$b_i = 2\sum_{k=1}^{K1} \alpha_k h(\theta_k, \mathbf{x}_i), \tag{20}$$

Above, $H = (1 + \exp(-\Theta\mathbf{X}))^{-1}$, where the exponential is applied element-wise. $H_{k,*}$ notation refers to a row vector corresponding to the k^{th} row respectively (k^{th} column for $H_{*,k}$), and tr is the matrix trace.

Regularisation. To prevent over-fitting of the model, we assume that the parameters have a Gaussian prior and constrain the diagonal inverse covariance matrix by a small number of hyper-parameters. We split the model parameters into three different groups: $\boldsymbol{\alpha}$, $\boldsymbol{\theta}$, and $[\boldsymbol{\beta}, \gamma]$, and assume that the parameters among different groups are independent of each other. This leads to the following log-likelihood function:

$$\begin{aligned} L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\Theta}) = \sum_{q=1}^{M} \log P(\mathbf{y}^{(q)} | \boldsymbol{x}^{(q)}) + \lambda_\alpha ||\boldsymbol{\alpha}|| \\ + \lambda_\beta ||\boldsymbol{\beta}|| + \lambda_\beta ||\boldsymbol{\gamma}|| + \lambda_\theta ||\boldsymbol{\Theta}|| \end{aligned} \tag{21}$$

The variances $\lambda_\alpha, \lambda_\beta, \lambda_\theta$ are determined automatically during model validation.

Inference. Since the CCNF model can be viewed as a multivariate Gaussian, inferring $\mathbf{y}'$ values that maximise $P(\mathbf{y}|\boldsymbol{x})$ (from Equation 1) is straightforward. The prediction is the mean (expected) value of the distribution:

$$\mathbf{y}' = \arg\max_{\mathbf{y}}(P(\mathbf{y}|\boldsymbol{x})) = \boldsymbol{\mu} = \Sigma \boldsymbol{d}. \tag{22}$$

Such straightforward and efficient inference is a major advantage of our model when compared to other structured prediction approaches, such as CNF.

Note that because of the sigmoid activation function, outputs y_i are limited the 0-1 range. However, most regression problems can be mapped to this range and later mapped back to their original ranges.

2.3 CCNF Instances

We propose two instances of the CCNF model that can exploit different output relationships. The first instance is called Chain-CCNF and can be used for time series and sequence modelling (illustrated in Figure 1a). The model has neighbouring nodes in time series connected using a single similarity edge feature:

$$S_{i,j}^{(l)} = \begin{cases} 1, |i-j| = 1 \\ 0, \text{otherwise} \end{cases} \tag{23}$$

This allows to capture the smoothness of the time-series signal. Even though, this example only connects neighbouring nodes it is easily extended to capture distant dependencies.

The second instance is called Grid-CCNF and it models spatial relationships. This model is illustrated in Figure 1b. Grid-CCNF is particularly suited for vision problems where regression across the image is needed, for example a probability density across an image. Sections 3 and 4 compare Chain- and Grid- CCNF to other popular regression models.

3 Chain-CCNF Experiments

We performed a set of experiments that explored the use of Chain-CCNF for time series modelling on two tasks: emotion prediction in music on the MTurk dataset [19] and facial action unit recognition on the DISFA dataset [20]. The following sections present the datasets, baselines, methodology and results.

3.1 Datasets

MTurk is a music extracts dataset labelled on the arousal-valence (AV) dimensional emotion space using Mechanical Turk [19]. Paid participants were asked to label 15-second excerpts with continuous emotion ratings on the AV space. The songs in the dataset cover a wide range of genres: pop, various types of rock and hip-hop/rap. MTurk dataset consists 240 15-second clips, with ≈ 17 ratings each - we used the ratings averaged over one second windows as ground truth. In addition, the dataset contains a standard set of features extracted from those musical clips. We used four types of features provided in the dataset: mel-frequency cepstral coefficients, chromagram, spectral contrast and statistical spectrum descriptors. They were concatenated into a single vector of 100 features per observation. Features were averaged over a one second window and their Z-scores were computed on the training set (same scalings used on the test set). For CCNF training the ground truth was scaled to [0,1], and the inverse scalings were used for testing.

DISFA - Denver Intensity of Spontaneous Facial Action database [20] contains 27, 4 minute long, videos of spontaneous facial expression, annotated for action units (AU) [21]. DISFA contains over 130,000 annotated frames from 27 adult participants. For every video frame, the intensity of 12 AUs was manually annotated on a six-point ordinal scale (0 and five increasing levels of intensity). AUs are a way to quantify facial muscle action and are a way to describe facial expressions. For our experiment these were treated as continuous values from 0 to 5 and not discrete classes.

For feature extraction we used an approach similar to the one proposed by Jeni *et al.*[7]. First, we extracted 25×25 pixel areas around facial feature points (provided with DISFA dataset). The areas of interest are also normalised for scale and rotation using affine transforms to frontal face and 32 pixel interocular distance. Only the areas around relevant feature points are extracted for each AU. For example for AU1 (inner brow raiser) prediction, we used the areas around eyebrow and eye-corner feature points. Each area of interest was normalised on a per subject basis to extract subject specific changes in appearance.

To reduce feature dimensionality of appearance features, we performed sparse non-negative matrix factorisation on each area of interest [22]. Unseen samples were projected on the non-negative basis using non-negative least squares (the basis was recomputed for each training fold). As an additional feature to appearance, we used the non-rigid shape parameters of a Point Distribution Model inferred from the provided landmark locations (the model was trained on the Multi-PIE dataset [23]).

Given the unbalanced nature of this dataset (most of the time neutral expression is shown), we rebalanced the training subset by keeping all sub-sequences with at least one AU activated and adding some temporal padding before and after. This lead to $\approx 40k$ frames per each AU with sequences of ≈ 100 frames. Note, that this rebalancing was only performed on the training set, for testing we used the original sequences. For CCNF training the ground truth was scaled to [0,1], and the inverse scalings were used for testing.

3.2 Baseline Models

We compared our Chain-CCNF model to the following approaches:

SVR which is a widely used model for regression. SVR treats each element in a sequence independently and does not perform structured regression. We explored both linear-kernel and radial basis kernel SVR.

Neural Network another popular regression technique. We used a version of CCNF model without edge features, which reduces to a neural network as time dependencies are not modelled any more. This allows us to evaluate the effect of the temporal CCNF features.

CCRF is a structured regression approach that can model temporal relationships [17,24]. The process of CCRF training requires an extra step, it first needs to train an SVR model to use its predictions as input to the CCRF model.

3.3 Methodology

For the **MTurk** dataset, a separate model was trained for each emotional dimension: one for arousal and one for valence. A 5-fold cross-validation testing was used for all of the experiments. When splitting the dataset into folds, it was made sure that all of the feature vectors from a single song were in the same fold. The reported test results were averaged over 5 folds. All experiments were performed on the same training and testing sets.

For hyper-parameter selection, 2-fold cross-validation (splitting into equal parts) was used on the training dataset. The automatically chosen hyper-parameters were used for training on the whole training dataset.

To evaluate the results, we used three metrics suggested for emotion prediction in music evaluation [25]: average Euclidean distance (combined dimensions), average root mean squared error and average squared Pearson correlation coefficient (per dimension).

We used leave-one-subject out testing in our experiments on the **DISFA** datasets, as done in previous work [20]. For hyper-parameter validation we used hold-out validation, with 2/3rds of the data used for training and the rest for validation for training all of the approaches.

To evaluate the results on the DISFA dataset, we used the Pearson correlation coefficient and RMSE across all test sequences (concatenating predictions from each test fold and then computing the error metrics).

3.4 Results and Discussion

CCNF consistently outperforms the baselines on most of the evaluation metrics on the MTurk dataset (Table 1). Not only is accuracy improved, but the results are substantially better than those of linear-kernel SVR, radial basis function SVR. Friedman's ANOVA ($\chi^2(3) = 49.6, p < 0.001$) on mean per-sequence Euclidean errors with follow up Wilcoxon tests on the Euclidean metric revealed that CCNF performs statistically significantly better than the other baselines.

Table 1. Results comparing the CCNF approach to the CCRF and SVR with linear and RBF kernels for emotion prediction on the MTurk dataset

Model	Arousal corr.	Valence corr.	Arousal rms	Valence rms	Euclidean distance
SVR-Lin	0.636	0.174	0.179	0.186	0.130
SVR-RBF	0.649	0.232	0.176	0.180	0.126
Neural Network	0.647	0.201	0.176	0.185	0.127
CCRF [24]	**0.744**	0.265	0.150	0.175	0.121
CCNF	0.732	**0.289**	**0.145**	**0.166**	**0.116**

Table 2. Results comparing the CCNF approach to other baselines on the DISFA dataset. The Pearson correlation metric and RMSE were computed on concatenated predictions across all of the 27 subjects. We also present a comparison against the best performing model from Sandbach *et al.*[12] (Built 1) for AUs reported in their paper.

Model	AU1	AU2	AU4	AU5	AU6	AU9	AU12	AU15	AU17	AU20	AU25	AU26	Avg.
	correlation												
SVR [7]	0.35	0.34	0.48	0.28	0.42	0.31	**0.71**	0.38	0.17	0.13	0.81	0.47	0.40
MRF [12]	**0.56**	**0.55**	0.44	0.17	0.14	0.01	-	-	-	-	-	-	-
CCRF [24]	0.44	0.37	0.43	0.40	0.38	0.26	0.65	0.35	0.31	**0.14**	0.77	0.46	0.41
CCNF	0.48	0.50	**0.52**	**0.48**	**0.45**	**0.36**	0.70	**0.41**	**0.39**	0.11	**0.89**	**0.57**	**0.49**
	rmse												
SVR [7]	0.74	0.70	1.09	**0.27**	0.78	0.64	**0.71**	0.41	0.59	**0.39**	0.81	0.67	0.65
MRF [12]	**0.62**	**0.59**	1.10	0.34	0.90	**0.61**	-	-	-	-	-	-	-
CCRF [24]	0.65	0.62	**1.04**	**0.27**	**0.72**	**0.61**	0.75	**0.39**	**0.57**	0.45	0.87	0.66	**0.63**
CCNF	0.74	0.63	1.13	0.33	0.75	0.67	**0.71**	0.46	0.67	0.58	**0.63**	**0.63**	0.66

For action unit recogntion the results of our experiment can be seen in Table 2. The CCNF model can be seen outperforming the other proposed baselines on most AUs for the correlation metric. Interestingly, CCNF does not do better on the RMSE metric than the baselines on some of the AUs while substantially outperforming the same baselines on the correlation metric (AU5, AU15, AU17). This is because RMSE is not a reliable metric for evaluating regressors on such an unbalanced dataset dominated by neutral expressions. This leads to an uninformed regressor that always predicts 0 performing well on RMSE metric.

None of the regressors are able to learn to recognise AU20 reliably. This is possibly because of two reasons: the features used are not discriminative enough, and there not enough positive training samples in the dataset – only 99 events for the action unit, when compared to 296 and 321 events for AU25 and AU26.

These results show the importance of modelling temporal and non-linear relationships for both emotion prediction in music and action unit recognition and the ability of CCNF to do so effectively.

4 Grid-CCNF Experiments

We conducted a set of experiments to evaluate our Grid-CCNF model as a patch expert in the constrained local model (CLM) – a popular framework for facial landmark detection [3,26]. Patch experts evaluate the probability of a landmark alignment at a particular pixel location and are a crucial part of CLM. The alignment probability is often modelled using a regressor trained to output 0 when landmark is not aligned and 1 when it is aligned. The evaluation of a patch expert in an area of interest leads to a *response map*. Landmark detection is done by optimising the CLM model over these response maps.

There has been a number of different patch experts proposed: various SVR models and logistic regressors, or even simple template matching techniques. The most popular patch expert by far is the linear SVR in combination with a logistic regressor [27,3,4]. Linear SVRs are used because of their computational simplicity and potential for efficient implementation on images using convolution.

There are some desirable properties for a response map from a patch expert[3,4]: the high probabilities should be be centred around the true landmark location, it should be smooth, preferably convex, and not have ambiguous peaks. Because of these desired properties, patch expert is a great use case for our Grid-CCNF model: (1) non-linear relationships between input pixel values and the output responses lead to better accuracy, and (2) ability to enforce similarity and sparsity relationships lead to a smoother and more convex response map.

To achieve desired similarity properties we define two similarity edge features: $S^{(g_1)}$ returns 1 (otherwise return 0) only when the two nodes i and j are direct (horizontal/vertical) neighbours in a grid; $S^{(g_2)}$ returns 1 (otherwise 0) when i and j are diagonal neighbours in a grid. For sparsity we define a single sparsity edge feature using the neighbourhood region $S^{(l)}$ that returns 1 only when two nodes i and j are between 4 and 6 edges apart (where edges are counted from the grid layout of our Grid-CCNF patch expert).

To train our Grid-CCNF patch expert, we need to define the output variables y_i. Given an image, with a true landmark location at $\mathbf{z} = (u, v)^T$, landmark alignment probability at $\mathbf{z}_i$ is modelled as $y_i = \mathcal{N}(\mathbf{z_i}; \mathbf{z}, \sigma)$ $(\sigma = 1)$.

4.1 Datasets

In order to evaluate the ability of Grid-CCNF to generalise on unseen datasets we evaluated our approach on four different datasets: Annotated Faces in the Wild (**AFW**)[28], **IBUG** [29], and 300 Faces in-the-Wild Challenge (**300-W**) [29] and **LFPW + Helen** [30,31] datasets. The datasets contain 135, 337, 600, and 554 images respectively. They all contain uncontrolled images of faces *in the wild*: in indoor-outdoor environments, under varying illuminations, in presence of occlusions, under different poses, and from different quality cameras. Note that LFPW and Helen were used as training datasets for CLM, CCNF, and AAM models, but same images were not used for testing.

4.2 Baseline Models

We used a number of state-of-the-art facial landmark detectors to compare to our CLM model that using Grid-CCNF patch experts:

CLM + SVR model that uses linear SVR patch experts and regularised landmark mean-shift fitting [3]. Same training data and initialisation was used for this model as for our CLM with Grid-CCNF. Note that we use a more accurate CLM model that that presented by Saragih *et al.*[3], our model includes a multi-modal (patch experts trained on raw pixel and gradient images) and multi-scale formulation leading to more accurate landmark detection.

Tree based face and landmark detector, proposed by Zhu and Ramanan [28]. It has shown good performance at locating the face and the landmark features on a number of datasets. We used a model trained on *in the wild* dataset [27].

DRMF – discriminative response map fitting implementation provided by the authors [27]. It was trained on LFPW [30] and Multi-PIE [23] datasets.

SDM – Supervised Descent Method implementation from the authors [32]. This approach is trained on the Multi-PIE and LFW [33] datasets. It relies on face detection from a Viola-Jones face detector.

AAM - Active Appearance Model using the inverse compositional image alignment algorithm [34]. This model was trained on *in the wild* data.

The above baselines were trained to detect 49, 66, or 68 feature points, making exact comparisons difficult. However, they all share 49 feature points, on which the error metrics were computed in our experiments.

4.3 Methodology

For Grid-CCNF and SVR patch expert training we used two datasets: Labelled Face Parts in the Wild (LFPW) [30] and Helen [31]. Both of them contain unconstrained images of faces in indoor and outdoor environments. In total 1029 training images were used, each of which was sampled at 6 locations (5 away from the true landmark location and 1 near it) in window sizes of 19×19 pixels - this led to 6 sequences of 81 samples per image. Each of the samples was an 11×11 pixel support region. Each training sample was normalised using their Z-score. For SVR each of the samples was treated independently. The models were trained using the LIBLINEAR package [35] with default parameters. For Grid-CCNF training the following parameters were used: 7 vertex features, $\lambda_\alpha = 10^2, \lambda_\beta = 10^3, \lambda_\Theta = 1$.

Nine sets of patch experts were trained in total: at three orientations – $-20°, 0°, 20°$ yaw; and three scales – 17, 23 and 30 pixel of interocular distance. Labels from the Helen and LFPW datasets were used to learn the point distribution model, using non-rigid structure from motion [36].

For fitting we used a multi-scale approach, with 15×15 pixel areas of interest for each scale. For model fitting we used regularised landmark mean-shift [3].

To initialise model fitting, we used the bounding boxes initialised using the tree based face detector [28]. In order to deal with pose variation the model was initialised at three orientations – $(0, 0, 0), (0, \pm 30, 0)$ degrees of roll, pitch and

Table 3. Results comparing CCNF patch expert to SVR patch expert on the LFPW and Helen datasets. Notice how CCNF can learn the relationship between input pixels and expected response map much better than SVR (all differences statistically significant according to a Wilcoxon signed-rank test, $p < .001$).

Scale	CCNF		SVR[3]	
	RMS	CORR.	RMS	CORR.
0.25	**0.083**	**0.33**	0.094	0.20
0.35	**0.086**	**0.27**	0.092	0.16
0.5	**0.089**	**0.22**	0.090	0.12
Avg.	**0.086**	**0.27**	0.092	0.16

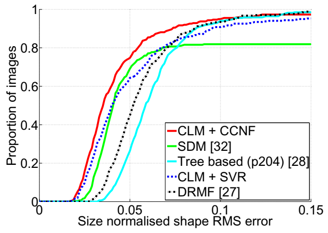
yaw. The converged model with the highest alignment likelihood was chosen as the correct one. Note that SDM code provided by the authors does not allow to use bounding box or rotation initialisation. hence the images were cropped to the face area to make them easier for the internal face detector (however, it still failed in more difficult cases).

4.4 Results and Discussion

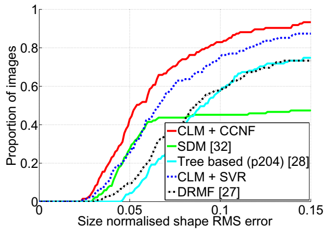
Before performing any landmark detection, we studied how CCNF patch experts compare to SVR ones at learning the expected patch response maps. We evaluated the patch experts on an unseen 1/5th of the images in the training datasets (LFPW and Helen). The results of this can be seen in Table 3. The results demonstrate that Grid-CCNF can learn the relationship between pixel values and expected alignment better than SVR at all of the training scales. The differences are statistically significant (see Table 3 caption).

The second experiment, compared the proposed CLM model that uses Grid-CCNF as a patch expert against other landmark detectors. The results of this experiment can be seen in Figure 2. Our approach can be seen outperforming all of the other baselines tested. The differences are statistically significant (see Figure 2 caption). This illustrates the greater generalisability of our proposed model over other approaches. This improved performance comes both from the learning capacity and the non-linearity of the neural network and the modelled sparsity and similarity constraints of our CCNF. Linear SVR is unable to accurately discriminate facial landmarks when their appearance varies due to large changes in illumination and pose.

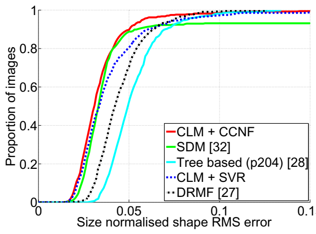
Training time for $68 \times 3 \times 3 = 612$ patch experts took ≈ 9 hours on a single quad core Intel i7 machine. We exploited the fact that patch experts of a profile face in one direction are just mirror images of a profile face in another direction, same fact was exploited for symmetric frontal face patch experts.



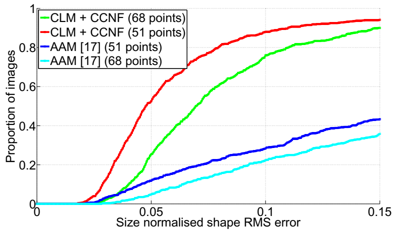
(a) AFW



(b) IBUG



(c) LFPW + HELEN



(d) 300-W

Fig. 2. Landmark detection using our **CLM + CCNF** model and other state-of-the-art methods. The benefit of our approach in generalising on unseen data is clear. Note that for 300-W dataset, error rates using just internal (51) and all of the points (68) are reported. CLM + CCNF has statistically significantly smaller RMS values compared to all of the other approaches on all of the datasets ($p < .001$), according to Friedman's ANOVA and Wilcoxon signed-rank follow up tests.

Finally, part of Grid-CCNF inference across an image area of interest can be performed using convolution, leading to fast landmark detection. Landmark detection speed depends on the area of interest, our CCNF implementation is able to reach 30 frames per second for landmark detection in videos and 5 images per second for more complex in the wild images.

5 Conclusions

We presented the Continuous Conditional Neural Field model for structured regression. Our model can exploit spatial and temporal relationships inherent in pattern recognition problems. It also learns non-linear dependencies between input and output. The flexibility of CCNF is demonstrated through evaluation on six datasets spanning three challenging regression tasks: landmark detection, emotion prediction in music and continuous intensity action unit recognition.

Our CCNF model showed statistically significant improvement over state-of-the-art approaches for all of the tasks, both in terms of prediction accuracy - evaluated using root mean squared error, and prediction structure - evaluated using the correlation coefficient. Finally, we make all of our code available for experiment recreation on the public datasets.

Acknowledgements. We acknowledge funding support from from the European Community's Seventh Framework Programme (FP7/2007- 2013) under grant agreement 289021 (ASC-Inclusion). The effort described here is partially supported by the U.S. Army. Any opinion, content or information presented does not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred.

References

1. Baltrusaitis, T., Morency, L.P., Robinson, P.: Constrained local neural fields for robust facial landmark detection in the wild. In: IEEE International Conference on Computer Vision Workshops (2013)
2. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer-Verlag New York, Inc. (2006)
3. Saragih, J., Lucey, S., Cohn, J.: Deformable Model Fitting by Regularized Landmark Mean-Shift. IJCV (2011)
4. Wang, Y., Lucey, S., Cohn, J.: Enforcing convexity for improved alignment with constrained local models. In: CVPR (2008)
5. Han, B.J., Rho, S., Dannenberg, R.B., Hwang, E.: Smers: Music emotion recognition using support vector regression. In: ISMIR (2009)
6. Valstar, M., Schuller, B., Smith, K., Eyben, F., Jiang, B., Bilakhia, S., Schnieder, S., Cowie, R., Pantic, M.: AVEC 2013 – The Continuous Audio / Visual Emotion and Depression Recognition Challenge (2013)
7. Jeni, L.A., Girard, J.M., Cohn, J.F., De La Torre, F.: Continuous au intensity estimation using localized, sparse facial feature space. In: FG (2013)
8. Nicolaou, M.A., Gunes, H., Pantic, M.: Output-associative RVM regression for dimensional and continuous emotion prediction. IVC (2012)
9. Wang, F., Verhelst, W., Sahli, H.: Relevance vector machine based speech emotion recognition. In: D'Mello, S., Graesser, A., Schuller, B., Martin, J.-C. (eds.) ACII 2011, Part II. LNCS, vol. 6975, pp. 111–120. Springer, Heidelberg (2011)
10. Sutton, C., McCallum, A.: Introduction to Conditional Random Fields for Relational Learning. In: Introduction to Statistical Relational Learning. MIT Press (2006)
11. Tsochantaridis, I., Hofmann, T., Joachims, T., Altun, Y.: Support vector machine learning for interdependent and structured output spaces. In: ICML (2004)
12. Sandbach, G., Zafeiriou, S., Pantic, M.: Markov random field structures for facial action unit intensity estimation. In: IEEE International Conference on Computer Vision, Workshop on Decoding Subtle Cues from Social Interactions (2013)
13. Peng, J., Bo, L., Xu, J.: Conditional neural fields. In: NIPS (2009)
14. Bo, L., Sminchisescu, C.: Structured output-associative regression. In: CVPR (2009)
15. Bo, L., Sminchisescu, C.: Twin gaussian processes for structured prediction. IJCV (2010)
16. Qin, T., Liu, T.Y., Zhang, X.D., Wang, D.S., Li, H.: Global ranking using continuous conditional random fields. In: NIPS (2008)
17. Baltrušaitis, T., Banda, N., Robinson, P.: Dimensional affect recognition using continuous conditional random fields. In: FG (2013)
18. Byrd, R.H., Lu, P., Nocedal, J., Zhu, C.: A limited memory algorithm for bound constrained optimization. SIAM Journal on Scientific Computing 16(5), 1190–1208 (1994)

19. Speck, J.A., Schmidt, E.M., Morton, B.G., Kim, Y.E.: A comparative study of collaborative vs. traditional musical mood annotation. In: ISMIR (2011)
20. Mavadati, S.M., Member, S., Mahoor, M.H., Bartlett, K., Trinh, P., Cohn, J.F.: Disfa: A spontaneous facial action intensity database. IEEE T-AFFC (2013)
21. Ekman, P., Friesen, W.V.: Manual for the Facial Action Coding System. Consulting Psychologists Press, Palo Alto (1977)
22. Kim, J., Park, H.: Toward faster nonnegative matrix factorization: A new algorithm and comparisons (2008)
23. Gross, R., Matthews, I., Cohn, J., Kanade, T., Baker, S.: Multi-pie. IVC 28(5), 807–813 (2010)
24. Imbrasaitė, V., Baltrušaitis, T., Robinson, P.: Emotion tracking in music using Continuous Conditional Random Fields and relative feature representation. In: IEEE International Conference on Multimedia and Expo (2013)
25. Imbrasaitė, V., Baltrušaitis, T., Robinson, P.: What really matters? a study into peoples instinctive evaluation metrics for continuous emotion prediction in music. In: Affective Computing and Intelligent Interaction (2013)
26. Martins, P., Caseiro, R., Henriques, J.F., Batista, J.: Discriminative bayesian active shape models. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part III. LNCS, vol. 7574, pp. 57–70. Springer, Heidelberg (2012)
27. Asthana, A., Zafeiriou, S., Cheng, S., Pantic, M.: Robust discriminative response map fitting with constrained local models. In: CVPR (2013)
28. Zhu, X., Ramanan, D.: Face detection, pose estimation, and landmark localization in the wild. In: IEEE CVPR (2012)
29. Sagonas, C., Tzimiropoulos, G., Zafeiriou, S., Pantic, M.: 300 faces in-the-wild challenge: The first facial landmark localization challenge. In: ICCV (2013)
30. Belhumeur, P.N., Jacobs, D.W., Kriegman, D.J., Kumar, N.: Localizing parts of faces using a consensus of exemplars. In: CVPR (2011)
31. Le, V., Brandt, J., Lin, Z., Bourdev, L., Huang, T.S.: Interactive facial feature localization. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part III. LNCS, vol. 7574, pp. 679–692. Springer, Heidelberg (2012)
32. Xiong, X., De la Torre, F.: Supervised descent method and its applications to face alignment. In: CVPR (2013)
33. Huang, G.B., Ramesh, M., Berg, T., Learned-Miller, E.: Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments (2007)
34. Matthews, I., Baker, S.: Active appearance models revisited. IJCV 60(2), 135–164 (2004)
35. Fan, R.E., Kai-Wei, C., Cho-Jui, H., Wang, X.R., Lin, C.J.: Liblinear: A library for large linear classification. The Journal of Machine Learning Research 9 (2008)
36. Torresani, L., Hertzmann, A., Bregler, C.: Nonrigid structure-from-motion: estimating shape and motion with hierarchical priors. TPAMI 30(5), 878–892 (2008)