CCNF FOR CONTINUOUS EMOTION TRACKING IN MUSIC: COMPARISON WITH CCRF AND RELATIVE FEATURE REPRESENTATION

Vaiva Imbrasaitė, Tadas Baltrušaitis, Peter Robinson

Computer Laboratory, University of Cambridge
Vaiva.Imbrasaite@cl.cam.ac.uk
Tadas.Baltrusaitis@cl.cam.ac.uk
Peter.Robinson@cl.cam.ac.uk

ABSTRACT

Whether or not emotion in music can change over time is not a question that requires discussion. As the interest in continuous emotion prediction grows, there is a greater need for tools that are suitable for dimensional emotion tracking. In this paper, we propose a novel Continuous Conditional Neural Fields model that is designed specifically for such a problem. We compare our approach with a similar Continuous Conditional Random Fields model and Support Vector Regression showing a great improvement over the baseline. Our new model is especially well suited for hierarchical models such as model-level feature fusion, which we explore in this paper. We also investigate how well it performs with relative feature representation in addition to the standard representation.

Index Terms— Music emotion recognition, dimensional representation, continuous tracking, machine learning

1. INTRODUCTION

Music surrounds us every day, and people's interaction with it is becoming increasingly digitized—buying digital music albums, streaming music, etc[1]. This introduces the need to develop better tools for music search, playlist generation and the general management of music libraries. People use a number of different descriptors for songs, and emotion has been shown to be one of them [2].

The field of emotion recognition in music started off focusing on assigning a single categorical label to an entire piece of music. Since then it has been slowly moving towards more complex techniques, with an increasing focus on dimensional representation of emotion, and continuous emotion tracking. Both of these, especially when combined, require more advanced machine learning techniques. However, until recently, only few of them have been used in the field (see Section 2).

In this paper we introduce a new Continuous Conditional Neural Fields (CCNF) model (Section 3.3) and apply it to the problem of continuous dimensional emotion tracking in music. We compare the performance of this model with Support Vector Regression (SVR), a standard baseline, and with a more advanced technique—continuous conditional random fields (CCRF). We show that our model outperforms the other two in most cases. We also explore the effect of using the relative feature representation and compare model-fusion with feature-fusion.

The code for CCNF and test-scripts that allow easy reproduction of our results are available on our website[1].

2. BACKGROUND

Dimensional emotion representation describes emotion using several axes. In the field of emotion in music, the most commonly used dimensions are arousal and valence (AV). Arousal describes how active/passive, and valence describes how positive/negative an emotion is. Adding other axes (such as expectancy, power) has also been considered, but it has repeatedly been shown that they add little, if anything, to the description or the recognition of emotion in music [3].

Most of the approaches to dimensional continuous emotion tracking have focused on inferring the emotion label over a time window, which is independent of the surrounding music (bag-of-frames approach) (Korhonen *et al.* [4], Panda and Paiva [5], Schmidt and Kim [6], Schmidt *et al.* [7], etc.). Since each second in a song is not independent from the music preceding it, this approach fails to exploit the temporal properties of music. Some research has been done on trying to incorporate temporal information in to the feature vector—either by using features extracted over varying window length for each second/sample [8], or by using machine learning techniques that are adapted for sequential learning (e.g. sequential stacking algorithm used by Carvalho and Chao [9], Kalman filtering or Conditional Random Fields (CRF) used by Schmidt and Kim [10, 11]).

[1]http://www.cl.cam.ac.uk/research/rainbow/projects/ccnf/

2.1. Continuous Conditional Random Fields

Continuous Conditional Random Fields have shown promise for continuous variable modeling when extra context is required (be it temporal or spatial information). They have been used to model emotions (both in music [12] and from human behaviour [13]), global ranking [14], remote sensing [15]. They are an extension of the Conditional Random Fields [16] to the continuous case, and have shown promise in these applications.

However, CCRFs face the problem that the model requires an initial prediction of the value being modeled, for example a local ranking for a global ranking problem, or a prediction from a single frame/time step regressor in the case of emotion tracking. The need to train multiple models complicates training, and might lead to a sub-optimal solution due to the regressors not being optimised jointly. Our approach attempts to ameliorate this problem.

2.2. Relative feature representation

Expectation has a substantial effect on our experience of listening to music. It is believed that violation of, or conformity to expectations when listening to music is a (main) source of musical emotion (proven by studies in neuroimaging [17], experimental aesthetics [18], etc.). Based on that, Imbrasaitė *et al.*[12] introduced a relative feature representation. They have shown that adjusting feature values relative to the average value of a song gives a substantial improvement over using the standard feature representation [19] and the improvement gained is similar to that of introducing more advanced machine learning techniques [12].

3. METHODOLOGY

3.1. Dataset

The dataset that we have used in our experiments is one of the few publicly available emotion tracking dataset of music extracts labeled on the arousal-valence dimensional space. The data [20] has been collected using Mechanical Turk (MTurk)[2], asking paid participants to label 15-second long excerpts with continuous emotion ratings on the AV space, with another 15 seconds given as a practice for each song. The songs in the dataset cover a wide range of genres—pop, various types of rock, hip-hop/rap, etc, and are drawn from the "uspop2002"[3] data-base containing popular songs. The dataset consists of 240 15-second clips (without the practice run), with 16.9 ± 2.7 ratings for each clip. In addition, the dataset contains a standard set of features extracted from those musical clips: MFCCs, octave-based spectral contrast, statistical spectrum descriptors (SSD), chromagram, and a set of EchoNest[4] features.

[2]https://www.mturk.com/ - accessed May 2013

[3]http://labrosa.ee.columbia.edu/projects/musicsim/uspop2002.html - accessed May 2013

3.2. Design of the experiments

For the purpose of this study, we only used the non EchoNest features provided in the MTurk dataset (Section 3.1). Features were averaged over a 1s window and the average of the labels for that second was used as the label. With all the machine learning models used, we trained two models separately—one for each axis.

3.2.1. Feature representation

Two types of **feature representations** are used in our work: basic and relative. For the basic feature representation, the features are concatenated into either a single vector (feature-level fusion) or into 4 separate vectors: MFCC, chromagram, spectral contrast and SSD (model-level fusion).

The relative feature representation we used in this paper was introduced by Imbrasaitė *et al.*[12, 19] (Section 2.2 for more information). Each (original) feature is represented by two numbers—the average of that feature over the whole duration of the song and the difference between the original value extracted from that frame and the average. As with the basic feature representation, both feature- and model-fusion were used.

3.2.2. Cross-validation

We used a 5-fold **cross-validation** for all the experiments. The dataset was split into two parts—4/5 for training and 1/5 for testing, and this process was repeated 5 times. When splitting the dataset into folds, we made sure that all of the feature vectors from a single song were in the same fold. The reported results were averaged over 5 folds.

For SVR-based experiments, we used a 2-fold cross validation (splitting into equal parts) on the training dataset to choose the hyper-parameters. These were then used for training on the whole training dataset.

The process for the CCRF-based experiments contained an extra step. The training dataset was split into two parts—one for SVR and one for CCRF, and we performed a 2-fold cross-validation on them individually to learn the hyper-parameters in the same way we do for the SVR-based experiments.

For CCNF-based experiments we again used a 2-fold cross validation to pick the hyper-parameters, but the results were averaged over 4 random seed initializations. We use the chosen hyper-parameters for training on the whole dataset—we randomly initialized the seed 20 times (using the best hyper-parameters) and picked the model with the highest likelihood (Eq.(6)) for testing.

[4]http://developer.echonest.com/ - accessed May 2013

It is important to note that the same folds were used for all of the experiments, and that the testing data was always kept separate from the training process.

3.3. Continuous Conditional Neural Fields

Our CCNF model, shown in Figure 1, brings the nonlinearity of Conditional Neural Fields [21] together with the flexibility and continuous output of Continuous Conditional Random Fields [14].

3.3.1. Model definition

We present an undirected graphical model that can model the conditional probability of a continuous valued vector $\mathbf{y}$ (for example the emotion in valence space) depending on continuous $\mathbf{x}$ (for example audio features).

In our discussion we will use the following notation: $\boldsymbol{x} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ is a set of observed input variables, $\mathbf{X}$ is a matrix where the i^{th} column represents $\boldsymbol{x}_i$, $\mathbf{y} = \{y_1, y_2, \dots, y_n\}$ is a set of output variables that we wish to predict, $\mathbf{x}_i \in \mathcal{R}^m$ and $y_i \in \mathcal{R}$ (patch expert response), n is the length of the sequence of interest.

Our model for a particular set of observations is a conditional probability distribution with the probability density function:

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

Above $\int_{-\infty}^{\infty} \exp(\Psi) d\mathbf{y}$ is partition function which forces the probability distribution to sum to 1.

We define two types of features in our model: vertex features f_k and edge features g_k. Our potential function is defined as:

$$\Psi = \sum_i \sum_{k=1}^{K1} \alpha_k f_k(y_i, \mathbf{x}_i, \boldsymbol{\theta}_k) + \sum_{i,j} \sum_{k=1}^{K2} \beta_k g_k(y_i, y_j) \tag{2}$$

In order to guarantee that our partition function is integrable [14] we constrain $\alpha_k > 0$ and $\beta_k > 0$, while $\boldsymbol{\Theta}$ is unconstrained. The model parameters $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \dots \alpha_{K1}\}$, $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots \boldsymbol{\theta}_{K1}\}$, and $\boldsymbol{\beta} = \{\beta_1, \beta_2, \dots \beta_{K2}\}$ are learned and used for inference during testing

The vertex features f_k represent the mapping from the $\mathbf{x}_i$ to y_i through a one layer neural network, where $\boldsymbol{\theta}_k$ is the weight vector for a particular neuron k.

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

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

The number of vertex features $K1$ is determined experimentally during cross-validation, and in our experiments we tried $K1 = \{5, 10, 20, 30\}$.

The edge features g_k represent the similarities between observations y_i and y_j. This is also affected by the neighborhood measure $S^{(k)}$, which allows us to control the existence of such connections.

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

In our linear chain CCNF model, g_k enforces smoothness between neighboring nodes. We define a single edge feature, i.e. $K2 = 1$. We define $S^{(1)}$ to be 1 only when the two nodes i and j are neighbors in a chain, otherwise it is 0.

3.3.2. Learning and Inference

We are given training data $\{\boldsymbol{x}^{(q)}, \mathbf{y}^{(q)}\}_{q=1}^{M}$ of M song samples, together with their corresponding dimensional continuous emotion labels. The dimensions are trained separately. In this section we describe how to estimate the parameters $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Theta}\}$, given the training data. It is important to note that all of the parameters are optimised jointly.

In learning, we want to pick the $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\Theta}$ values that optimise the conditional log-likelihood of our model on the training sequences:

$$L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Theta}) = \sum_{q=1}^{M} \log P(\mathbf{y}^{(q)} | \boldsymbol{x}^{(q)}) \tag{6}$$

$$(\bar{\boldsymbol{\alpha}}, \bar{\boldsymbol{\beta}}, \bar{\boldsymbol{\Theta}}) = \underset{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Theta}}{\arg\max} (L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Theta})) \tag{7}$$

Similarly to Baltrušaitis *et al.* [13] and Radosavljevic *et al.* [15], we convert the Eq.(1) into multivariate Gaussian form. It helps with the derivation of the partial derivatives of log-likelihood, and with the inference.

$$P(\mathbf{y}|\boldsymbol{x}) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} \exp(-\frac{1}{2}(\mathbf{y} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{y} - \boldsymbol{\mu})), \tag{8}$$

$$\Sigma^{-1} = 2(A + B) \tag{9}$$

The diagonal matrix A represents the contribution of α terms (vertex features) to the covariance matrix, and the symmetric B represents the contribution of the β terms (edge features). They are defined as follows:

$$A = (\sum_{k=1}^{K1} \alpha_k) I \tag{10}$$

$$B_{i,j} = \begin{cases} (\sum_{k=1}^{K2} \beta_k \sum_{r=1}^{n} S_{i,r}^{(k)}) - (\sum_{k=1}^{K2} \beta_k S_{i,j}^{(k)}), & i = j \\ -\sum_{k=1}^{K2} \beta_k S_{i,j}^{(k)}, & i \neq j \end{cases} \tag{11}$$

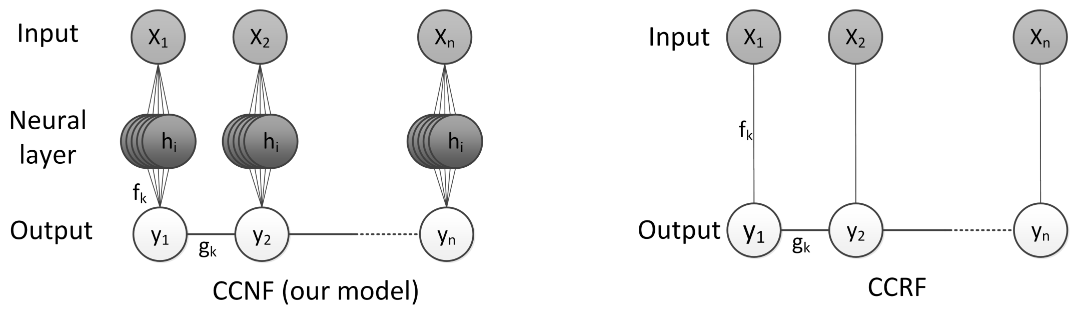

Fig. 1. Our linear-chain CCNF model compared with the linear-chain CCRF one [12, 13]. The input vector $\mathbf{x}_i$ is connected to the relevant output scalar y_i through the vertex features that combine the h_i neural layers (gate functions) and the vertex weights $\boldsymbol{\alpha}$. The outputs are further connected with edge features g_k

We also define $\boldsymbol{\mu} = \Sigma \boldsymbol{d}$ which is the expected (mean) value of the Gaussian CCNF distribution.

$$\boldsymbol{\mu} = \Sigma \boldsymbol{d} \tag{12}$$

It is defined in terms of $\Sigma = (2A + 2B)^{-1}$, and a vector $\boldsymbol{d}$, that describes the linear terms in the Gaussian distribution:

$$\mathbf{d} = 2\alpha^T h(\Theta \mathbf{X}) \tag{13}$$

Above Θ represents the combined neural network weights and $h(\Theta\mathbf{X})$, is an element-wise application of h on each element of the resulting matrix. Intuitively $\boldsymbol{d}$ is the contribution from the the vertex features towards $\mathbf{y}$.

For learning we can use the constrained L-BFGS for finding locally optimal model parameters. We use the standard Matlab implementation of the algorithm. In order to make the optimisation both more accurate and faster we used the partial derivatives of the $\log P(\mathbf{y}|\mathbf{x})$, which are straightforward to derive and are similar to those of CCRF [13] (details omitted here for brevity).

In order to avoid overfitting we add L_2 norm regularisation terms to the likelihood function for each of the parameters types ($\lambda_\alpha ||\boldsymbol{\alpha}||_2^2$,$\lambda_\beta ||\boldsymbol{\beta}||_2^2$, $\lambda_\theta ||\boldsymbol{\Theta}||_2^2$). The values of $\lambda_\alpha, \lambda_\beta, \lambda_\Theta$ are determined during crossvalidation, as is the number of neural layers.

For inference we need to find the value of $\mathbf{y}$ that maximises $P(\mathbf{y}|\boldsymbol{x})$. Because $P(\mathbf{y}|\boldsymbol{x})$ can be expressed as a multivariate Gaussian distribution (Eq.(8)) , the value of $\mathbf{y}$ that maximises it is the mean value of the distribution, hence:

$$\mathbf{y}^* = \arg\max_{\mathbf{y}} (P(\mathbf{y}|\boldsymbol{x})) = \boldsymbol{\mu} \tag{14}$$

3.4. Feature-level and Model-level fusion

Our CCNF model can accommodate different types of feature fusion. For feature fusion we just use single vector $\mathbf{x}_i$ containing all of the different modality features. For model-level fusion we split the vector into different modalities leading to more feature functions, where each modality has its own corresponding f_k's. We define separate vertex functions for MFCC, chromagram, spectral contrast and SSD features.

For SVR-based model-level fusion, we train 4 separate models—one for each class of features. Then a feature vector composed of the predicted labels only is used for the final model.

Table 1. Results comparing our CCNF approach to the CCRF and SVR with linear and RBF kernels.

Experiment	Ar.		Val.		Eucl.
	RMS	Corr	RMS	Corr	
SVR-Lin	0.196	0.634	0.222	0.173	0.130
SVR-RBF	0.194	0.645	0.220	0.211	0.128
CCRF [12]	0.204	0.721	0.223	0.247	0.136
CCNF	**0.166**	**0.739**	**0.205**	**0.301**	**0.116**

Table 2. Results (using short correlation and RMS) comparing our CCNF approach to the CCRF and SVR with linear and RBF kernels.

Experiment	Ar.		Val.	
	RMS	Corr	RMS	Corr
SVR-Lin	0.180	0.012	0.189	0.036
SVR-RBF	0.178	0.011	0.186	0.007
CCRF [12]	0.176	0.049	0.183	**0.090**
CCNF	**0.143**	**0.072**	**0.170**	0.019

4. RESULTS

We use 3 different evaluation metrics for all of our experiments: correlation, root-mean-square error (RMSE) and Euclidean distance. Both the correlation coefficient and RMSE

Table 3. Results comparing our CCNF approach to the CCRF and SVR with linear and RBF kernels using relative feature representation.

Experiment	Ar.		Val.		Eucl.
	RMS	Corr	RMS	Corr	
SVR-Lin	0.169	0.728	0.220	0.160	0.126
SVR-RBF	**0.167**	**0.735**	0.209	**0.297**	**0.117**
CCRF [12]	0.179	0.718	0.216	0.257	0.123
CCNF	**0.167**	0.733	**0.207**	0.281	0.120

Table 4. Results (using short correlation and RMS) comparing our CCNF approach to the CCRF and SVR with linear and RBF kernels using relative feature representation.

Experiment	Ar.		Val.	
	RMS	Corr	RMS	Corr
SVR-Lin	0.145	0.013	0.188	0.026
SVR-RBF	**0.143**	0.046	0.170	0.035
CCRF [12]	0.153	**0.071**	0.176	0.049
CCNF	0.145	0.058	**0.169**	**0.064**

are calculated in two modes, which we call short and long. Long evaluation metrics are calculated over the span of the whole dataset, essentially concatenating all of the songs into one. Short evaluation metrics are calculated over each song and then averaged over all of the songs. The short correlation we report is non-squared, so as not to hide any potential negative correlation. We believe that short metrics are better suited for evaluation of emotion recognition in music, as the performance of our algorithms on a given song is what actually matters. We report long metrics since these are usually reported in the literature. The average Euclidean distance is calculated as the distance between the two-dimensional position of the original label and our predicted labels in the normalized AV space (each axis normalized to span between 0 and 1). Each metric is calculated for each fold and the average over 5 folds is reported.

Please note that lower RMSE and Euclidean distance values correspond to better performance, while the opposite is

Table 5. Results comparing our CCNF approach to the SVR with RBF kernels using model-level fusion, basic (B) and relative (R) representation.

Experiment	Ar.		Val.		Eucl.
	RMS	Corr	RMS	Corr	
SVR-B	0.205	0.632	0.216	0.207	0.129
SVR-R	0.186	0.714	**0.204**	**0.304**	0.123
CCNF-B	**0.168**	**0.737**	0.209	0.263	**0.118**
CCNF-R	0.172	0.722	0.226	0.183	0.125

Table 6. Results (using short correlation and RMS) comparing our CCNF approach to the SVR with RBF kernels using model-level fusion, basic (B) and relative (R) representation. relative-short

Experiment	Ar.		Val.	
	RMS	Corr	RMS	Corr
SVR-B	0.182	-0.003	0.182	0.058
SVR-R	0.159	-0.001	**0.167**	0.032
CCNF-B	**0.146**	**0.043**	0.171	**0.073**
CCNF-R	0.148	-0.014	0.183	0.001

true for correlation.

4.1. Standard feature representation

CCNF with the basic feature representation consistently outperforms all of the other methods in all the evaluation metrics except for short correlation for valence, where CCRF performs better (Tables 1 and 2). Not only is the performance improved, but it can be seen that the results achieved with CCNF are substantially better than those of the other methods.

4.2. Relative feature representation

The results with the relative feature representation are less consistent (Tables 3 and 4). Even though CCNF clearly outperforms CCRF and SVR with the linear kernel, the results are nearly identical to those achieved by the SVR model with the RBF kernel. That is especially evident with long, rather than short evaluation metrics.

4.3. Model-level fusion

Model-level fusion does not appear to have a strong positive effect, and it varies greatly depending on which machine learning technique is used and which axis we are working with. For valence, SVR model using relative feature representation and the RBF kernel seems to achieve one of the best results (Table 5). For arousal, CCNF with basic representation seems to be performing the best.

5. DISCUSSION

In this paper we introduced a new CCNF model that is particularly well suited for dimensional continuous emotion tracking. In addition to that, the model can easily incorporate multi-modal features, or implement model-level fusion.

The results we achieved with this model are very encouraging. With the basic feature representation, it consistently outperformed the other models we compared it to—both the standard baseline used in the field (SVR) and the more advanced CCRF model. With the relative feature representation, the results are more mixed—our model definitely outperforms CCRF, but the performance is often matched by SVR with the RBF kernel.

We have used the Euclidean distance in order to compare our work to that of the state of the art. Schmidt *et al.*[10] used the same dataset for their experiments that were based on a similar experimental design. They reported mean Euclidean of 0.160-0.169 which is within the same order of magnitude as the best average Euclidean distance of 0.116 we achieved in our experiments. This suggests that our model is at least of a comparable quality as the state of the art and can most likely achieve better performance.

We found the model-level fusion did not give the improvement we expected. It seems that the improvement is larger when a simpler machine learning technique is used. This leads us to the hypothesis that there is a balance between the complexity of a machine learning method used, and the complexity of the feature vectors used. It seems that when using simpler machine learning techniques, the results can be greatly improved by spending more time carefully designing the features employed. On the other hand, when a more advanced method is used we get diminishing return from more complex feature vectors. It would therefore appear that the relationships we uncover by building complex features can also be implicitly learned with more advanced techniques.

6. REFERENCES

[1] BPI, "Digital Music Nation," Tech. Rep., 2013.

[2] D Bainbridge, S J Cunningham, and J S Downie, "How People Describe Their Music Information Needs : A Grounded Theory Analysis Of Music Queries," in *Proc. of ISMIR*, 2003, pp. 221–222.

[3] K F MacDorman and Stuart Ough Chin-Chang H, "Automatic Emotion Prediction of Song Excerpts: Index Construction, Algorithm Design, and Empirical Comparison," *Journal of New Music Research*, vol. 36, no. 4, 2007.

[4] M D Korhonen, D A Clausi, and M E Jernigan, "Modeling emotional content of music using system identification," *IEEE transactions on systems man and cybernetics Part B Cybernetics*, vol. 36, no. 3, 2006.

[5] R Panda and R P Paiva, "Using Support Vector Machines for Automatic Mood Tracking in Audio Music," in *130th Audio Engineering Society Convention*, 2011.

[6] E M Schmidt and Y E Kim, "Prediction of time-varying musical mood distributions from audio," in *Proc. of ISMIR*, 2010, pp. 465–470.

[7] E M Schmidt, D Turnbull, and Y E Kim, "Feature selection for content-based, time-varying musical emotion regression," in *Proc. of ISMIR*. 2010, ACM.

[8] E Schubert, "Modeling Perceived Emotion With Continuous Musical Features," *Music Perception*, vol. 21, no. 4, 2004.

[9] V R Carvalho and Chih-yu Chao, "Sentiment Retrieval in Popular Music Based on Sequential Learning," *Proc. ACM SIGIR*, 2005.

[10] E M Schmidt and Y E Kim, "Prediction of Time-Varying Musical Mood Distributions Using Kalman Filtering," *9th ICMLA*, pp. 655–660, 2010.

[11] E M Schmidt and Y E Kim, "Modeling musical emotion dynamics with Conditional Random Fields," *Proc. of ISMIR*, 2011.

[12] V Imbrasaitė, T Baltrusaitis, and P Robinson, "Emotion tracking in music using Continuous Conditional Random Fields and relative feature representation," in *Proc. IEEE ICME*, 2013.

[13] T Baltrušaitis, N Banda, and P Robinson, "Dimensional Affect Recognition using Continuous Conditional Random Fields," in *IEEE FG*, 2013.

[14] T Qin, T Liu, X Zhang, D Wang, and H Li, "Global Ranking Using Continuous Conditional Random Fields," in *NIPS*, 2008.

[15] V. Radosavljevic, S. Vucetic, and Z. Obradovic, "Continuous Conditional Random Fields for Regression in Remote Sensing," in *ECAI*, 2010, pp. 809–814.

[16] C Sutton and A Mccallum, *Introduction to Conditional Random Fields for Relational Learning*, MIT Press, 2006.

[17] S Koelsch, W A Siebel, and T Fritz, "Chapter 12, Functional neuroimaging," in *Handbook of music and emotion theory research application*, P N Juslin and J A Sloboda, Eds., pp. 313–346. OUP, 2010.

[18] D J Hargreaves and A C North, "Experimental aesthetics and liking for music," in *Handbook of music and emotions theory research applications*, P N Juslin and J A Sloboda, Eds., chapter 19, pp. 515–547. OUP, 2010.

[19] V Imbrasaitė and P Robinson, "Absolute or relative? A new approach to building feature vectors for emotion tracking in music," in *Proc. ICME3*, 2013.

[20] J A Speck, E M Schmidt, B G Morton, and Y E Kim, "A comparative study of collaborative vs. traditional music mood annotation," *Proc. of ISMIR*, 2011.

[21] J Peng, L Bo, and J Xu, "Conditional neural fields," *Advances in Neural Information Processing Systems*, vol. 22, 2009.