Proceedings of the Federated Conference on Computer Science and Information Systems pp. 53–60
DOI: 10.15439/2016F309
ACSIS, Vol. 8. ISSN 2300-5963

Sparse Coding Methods for Music Induced Emotion Recognition

Jan Jakubik
Wrocław University of Science and Technology
Department of Computational Intelligence
Wrocław, Poland
Email: jan.jakubik@pwr.edu.pl

Halina Kwaśnicka
Wrocław University of Science and Technology
Department of Computational Intelligence
Wrocław, Poland
Email: halina.kwasnicka@pwr.edu.pl

***Abstract*—The paper concerns automatic recognition of emotion induced by music (MER, Music Emotion Recognition). Comparison of different sparse coding schemes in a task of MER is the main contribution of the paper. We consider a domain-specific categorization of emotions, called Geneva Emotional Music Scale (GEMS), which focuses on induced emotions rather than expressed emotions. We were able to find only one dataset, namely Emotify, in which data are annotated with GEMS categories, this set was used in our experiments. Our main goal was to compare different sparse coding approaches in a task of learning features useful for predicting musically induced emotions, taking into account categories present in the GEMS. We compared five sparse coding methods and concluded that sparse autoencoders outperform other approaches.**

I. Introduction

MUSIC information retrieval has been gaining an increased amount of attention in machine learning community over the past decade due to the increasing popularity of new musical services and a huge amount of music data available on the internet, annotated using imperfect systems such as community tagging and description supplied by producers. With the increasing amount of files, automated music analysis and content-based retrieval becomes increasingly relevant.

One particular subset of music information retrieval tasks is music emotion recognition (MER) [1]. Emotion recognition, besides the potential use in recommendation systems and search engines, poses an interesting theoretical problem for artificial intelligence researchers due to its high subjectivity and the fact human emotions are still not fully understood by psychologists. For music emotion recognition, even the set of emotions employed to annotate the dataset and use as a ground truth in machine learning is still subject to discussion. Emotional categories and scales in popular datasets are often not domain-specific.

Another important issue in automated emotion recognition is that emotion can often be related to musicological concepts such as chords and tempo [2], which from a machine learning standpoint correspond to high-level features that require specialized algorithms to extract them from the music files. However, these algorithms can be imperfect and affect the performance significantly. Meanwhile, current trends in deep learning suggest that algorithms which learn to extract more complicated features from low-level features in an unsupervised manner can achieve even better results than hand-crafted features designed for a specific domain [4]. In music information retrieval, this kind of approach has become popular in the context of genre recognition [5][6] and generalized auto-tagging task [7][8], which includes simple emotional tags, but is more focused on semantic categories such as genre, male/female vocalist, etc.

In this paper, we describe a machine learning approach based on aggregation of sparse vectors constructed from the low-level description of a sound file and apply it to Emotify, a publicly available MER dataset. It is different from usual emotion recognition datasets in that it uses Geneva Emotional Music Scale, an emotional scale designed specifically for music-induced emotion. We compare different sparse coding approaches in order to find out whether these methods can reliably learn features useful for recognition of musically induced emotion, especially the highly subjective and hard to define emotional categories present in GEMS, for which the usually employed emotional scale with Valence-Arousal dimensions does not account.

The paper is organized as follows: Section II summarizes literature related to our paper. Section III describes GEMS emotion categorization and the dataset we used in our experiments. Section IV describes different sparse coding methods which can be used interchangeably in our approach. Section V explains our representation of music files, create using sparse vector pooling, and the rationale behind it. In Section VI, we report the results of our experiments on the Emotify dataset.

II. Related work

Music emotion recognition is an important part of automated music information retrieval and as such, it has been covered extensively in existing literature. Typically, an MER task in artificial intelligence research takes one of two forms: a multi-class annotation task in which each song in a dataset is annotated with a subset of a predefined set of tags [9][10], or a regression task multiple continuous values representing different dimensions of perceived emotion are assigned to a song [11][12]. In the case of multi-class annotation, there is no single consistent system of emotional categories. The set of tags can be based on one of existing emotion categorization systems in psychology [13] [14]. However, these are not domain-specific and can be imperfect for describing music.

A common concept in work focusing on dimensional scales is the Valence-Arousal space [15], in which only two continuous dimensions are considered. Valence differentiates between positive and negative emotions while Arousal describes the intensity of perceived emotion. E.g., music described as "energetic" or "joyful" could be placed on the V-A scale in the high valence, high arousal area while music described as "calm" will be in the low arousal, middle valence area. It should be noted that the concept of placing specific emotions on the V-A scale was not designed specifically for describing emotions concerning music. Another criticism of this scale is that its two dimensions are insufficient to describe more complex emotions and discern pairs of emotions such as fear and anger (both of which are negative emotions with high arousal). The question whether V-A scale should be extended with a third dimension is an important subject of discussion among both MIR researchers and psychologists [16].

An important aspect of music emotion is the differentiation between *induced* and *expressed* emotions. Since one of the basic purposes of music is influencing the listener's mood, that differentiation should be a subject of interest when researching emotion modelling. A domain-specific emotion categorization system called Geneva Emotional Music Scale which takes the distinction between induced and expressed emotion into account was proposed in [17]. We describe it in more detail in section III, along with Emotify, the first MER dataset annotated with GEMS categories. The dataset was researched from a machine learning viewpoint in [18], which examined the performance of available features from multiple existing MATLAB toolboxes in combination with SVR algorithm. However, more complicated approaches such as stochastic process modelling or codebook methods have not been applied to GEMS-annotated data yet.

Among existing music information retrieval approaches, we chose to examine codebook methods. Codebook methods rely on building a *dictionary* of discernible patterns appearing in sample data on a short time-scale and then expressing the contents of a music file using contents of the dictionary. In recent years, papers concerning codebook methods reported good results in music information retrieval tasks [5][8] using the Restricted Boltzmann Machine algorithm [19]. In [8], results on the state of the art level were achieved in generalized music auto-tagging using temporal pooling of sparse vectors to represent a music file. We apply the same pooling approach to music emotion modelling. The authors established that sparse RBM can learn features better than sparse coding methods used for codebook generation in the past. Another promising coding method that, unlike RBM, has not been tested extensively in this context is an autoencoder neural network. Autoencoders without sparsity constraints have been tested as an emotion recognition method in [20], using a separate network for each modelled emotion. They have been applied to chord recognition [21] with good results. Hence, in our experiments we test autoencoder with a sparsity inducing loss function as an alternative to RBM.

III. GEMS Emotion Recognition

Geneva Emotional Music Scale is a categorical model of emotion designed specifically for the music domain, based on psychological research [17]. GEMS authors propose a hierarchy with three levels, with three general categories on the top level, nine emotions in the middle and 45 specific emotions at the bottom. The emotional categories were created using surveys concerning music-induced emotion. It is important to note that the difference between emotions expressed by music and induced by music is relevant to the choice of terms. Surveys explicitly asked separate questions about emotions participants felt and emotions they perceived in music.

In [22], nine emotions from the middle level of GEMS hierarchy were chosen to annotate the Emotify dataset, consiting of 400 songs from 4 genres. These middle-level categories are: amazement, solemnity, tenderness, nostalgia, calmness, power, joyful activation, tension, and sadness. The annotations were gathered using a Facebook game Emotify, and thus can represent the task of modelling a community consensus. For each song, annotations in the form of vectors consisting of zeroes (emotion not felt) and ones (emotion felt) were gathered from multiple subjects. Overall annotation of a song can be calculated as a mean of the annotation vectors corresponding to it, resulting in a vector of 9 continuous values ranging from 0 to 1, where zero means a complete agreement that the song does not evoke a particular feeling and one complete agreement that it evokes said feeling.

The dataset was analysed from a machine learning viewpoint in [18]. Authors compared three different sets of automatically extracted features and a set of manually annotated musicological features. First analysed feature set was features designed for music description available in MIRtoolbox, a MATLAB toolbox for music information retrieval. The second one used Mel-frequency Cepstral Coefficients (MFCC) [23] and statistical functionals applied to them such as mean, variance, skewness, etc. Third feature set consisted of harmonic features proposed by the authors used in combination with MIRtoolbox features. The authors concluded that while certain emotions can be modelled with decent accuracy, the performance of machine learning is heavily limited by the issue of subjectivity. The most subjective emotional categories identified by the authors were amazement, solemnity, and tension.

IV. Sparse Representation of Data

Sparse coding is the idea of approximating data drawn from a multidimensional space by representing it in another space in a way that encourages sparsity, i.e. a vector in the output space should consist mostly of zeroes. In its simplest form, the problem of approximating a vector y with its sparse representation x in a base D (called a dictionary), can be defined as:

$$x^* = \arg\min_x \|x\|_0 \quad s.t. \quad \|y - Dx\| < \lambda \tag{1}$$

where λ is a parameter dictating the desired accuracy of reconstruction and $\|x\|_0$ is the number of non-zero elements of vector x. However, this optimization problem is NP-hard and assumes a linear transformation between two spaces, which can sometimes be insufficient to create sparse representations for complex data. Below we describe approaches practically applicable to the problem of sparse representation for any given set of data.

A. K-means Clustering

K-means clustering [24] is a well-known unsupervised learning algorithm in which a dataset is divided into clusters, and each cluster is represented by a single point in the data space, which is the centroid of the cluster. The algorithm assigns any vector in the dataset to the cluster represented by a point closest to it. This enables us to treat K-means clustering as a very restrictive form of sparse coding. A vector y in the original data space assigned to n-th cluster can be represented by a vector x in which every component except n-th is 0, and the n-th component is 1. The dimensionality of x is equal to the number of clusters. Note that this is equivalent to minimizing $\|y - Dx\|$ with constraints $\|x\|_0 = 1$ and $\|x\|_1 = 1$, where the dictionary D is a matrix which n-th column is the centroid of the n-th cluster.

For any vector which is not present in the dataset used for dictionary training, its sparse representation can be calculated by simply choosing the centroid closest to it, and creating a vector of zeros with n-th component being one, where n is the closest centroid. This allows us to express the popular Bag-of-Words model [3] as a specific case of our approach, which will be explained in section IV. This model is known for its simplicity and efficiency regarding the dictionary building process.

B. L1 Regularized Least Squares

A basic way of solving the sparse representation problem in polynomial time is to use L1 norm regularization:

$$x^* = \arg\min_x \|y - Dx\| + \lambda\|x\|_1 \tag{2}$$

L1 norm is known to encourage sparsity [25] and there has been a significant amount of research dedicated to fast solvers for both L1 regularized least squares and its nonnegative variant. Nonlinearity is possible to achieve using the kernel trick [26], the problem then becomes:

$$x^* = \arg\min_x \|\Phi(y) - \Phi(D)x\| + \lambda\|x\|_1 \tag{3}$$

where Φ denotes a mapping function from the original data space to a space of larger dimensionality, implicitly defined by a kernel function $K(a, b)$ which replaces dot product.

Variants of regularized least squares are commonly used in sparse coding problems. They can achieve better reconstruction than a K-means based dictionary approach, at the cost of more complicated dictionary building process. Another issue is that the encoding of a vector is not explicitly given and requires solving (2), which is the main disadvantage of this approach.

C. Sparse Autoencoder

Autoencoders [27] are a type of neural networks developed mostly for use in deep learning. In its basic form, an autoencoder is simply a standard feedforward neural network in which the number of input neurons is equal to the number of output neurons. The network is trained using a backpropagation algorithm [28], in which the network is given a matrix of training vectors X and a matrix desired outputs Y. In the case of autoencoders $Y = X$, meaning the objective of learning is to create an encoding in hidden layer that enables the network to reconstruct later input vectors with maximum possible accuracy.

Without additional modifications to a standard loss function used in backpropagation (e.g. mean squared error), data compression is achieved by using a hidden layer with a low number of neurons, relatively to the number of input neurons. However, it is possible to encourage sparsity [29] by modifying the loss function for hidden layer. Denoting a loss function in relation to data matrix X and desired output matrix Y as $L(X, Y)$, the new loss function minimized by the backpropagation algorithm becomes:

$$L'(X, Y) = L(X, Y) + \lambda \sum_i (\rho \log \frac{\rho}{\hat{\rho_i}} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho_i}}) \tag{4}$$

where ρ is a parameter indicating the desired average activations in the hidden layer, and $\hat{\rho_i}$ is the average activation of i-th neuron in that layer over the whole dataset. This encourages neuron activations in hidden layer to be 'sparse' in the sense that only a few neurons relevant to recognizing a particular data pattern are 'active' in response to a given input, i.e. they should exhibit significantly higher activations than all the others. However, the vector of hidden layer outputs is not sparse in a strict sense, as most of the 'inactive' neuron outputs are not equal to 0, and it is nearly impossible to achieve zero activations through backpropagation. We will explain why this is not an issue in our model when discussing the representation of music by pooling sparse vectors.

D. Sparse RBM

Restricted Boltzmann Machine [19] is a stochastic neural network model consisting of a set of visible units and a set of hidden units. These layers are fully connected to each other, however, there are no connections inside a layer. Assuming Gaussian visible nodes and binary hidden nodes (which is sufficient to model real-valued inputs) a configuration of network (x, h), where x is the vector of visible values and y is the vector of hidden values, is associated with an energy function:

$$E(x, h) = \frac{1}{2\sigma^2} x^T x - \frac{1}{\sigma^2} x^T W h - a^T x - b^T h \tag{5}$$

where W is the weight matrix, a and b are bias terms. σ is a scaling parameter. This energy function is inversely proportional to the log-likelihood of observing a particular configuration. Probability distribution of $x^{(i)}$, the value of i-th visible neuron, assuming a hidden layer configuration h is Gaussian with mean $a_i + w_i^T h$ and standard deviation σ:

$$p(x^{(i)}|h) = \mathcal{N}(a_i + w_i^T h, \sigma^2) \tag{6}$$

where w_i denotes the i-th row of W and a_i the i-th component of bias vector a. Probability for j-th hidden layer unit being active is given by:

$$p(h^{(j)}|x) = sig(\frac{1}{\sigma^2}(b_j + w_j x)) \tag{7}$$

where b_j is the j-th component of bias vector b, w_j is the j-th column of W and sig is the logistic sigmoid function. Given these definitions, a model of (W, a, b) can be learned by maximizing the log-likelihood of visible values x using the contrastive divergence learning [30].

A penalty term can be added to the objective function to encourage sparsity. Given a sequence of training vectors in which i-th vector is x_i, the penalty term is:

$$penalty = \lambda \sum_j (\rho - \frac{1}{m} \sum_i \mathbf{E}(h^{(j)}|x_i)) \tag{8}$$

where ρ is the desired average activation of hidden layer and λ is a scaling parameter.

Encoding of a given input vector can be calculated using equation (7), which is similar to coding in autoencoder networks. Restricted Boltzmann Machines are known for their efficiency in building deep neural network architectures, however, these deep networks are usually fine-tuned after the initial learning, using backpropagation. In our approach, this fine-tuning in a supervised manner is not possible.

V. Music Representation by Pooling Sparse Vectors

We apply a method of representing music files described in [8], in which a machine learning scheme based on the sparse representation of data was employed to great success. Proposed approach achieved state-of-art results in the generalized music annotation task, in which the goal is to select a subset from a set of predefined tags for each music file in the dataset. These tags included genre tags, simple categorical emotion tags and other descriptive tags such as 'female vocalist'. It is important to note that emotion recognition does not necessarily rely on the same features of music as genre recognition. For emotion recognition, it is significantly more important to take rhythmic, tempo and harmonic features into account. For example, the differentiation between major and minor chords, while not very important when differentiating between classical and rock music, becomes crucial when attempting to differentiate sad and happy songs due to strong cultural associations between major-minor scales and emotions of happiness-sadness.

In our approach, firstly we calculate the spectrogram of a music file, using log scale for frequency. A sequence of vectors is built in which every vector represents a spectrogram patch of l consecutive frames of the spectrogram, with maximum overlap between patches (i.e. for a window length l a vector x_i in the sequence contains frames from i to $i+l-1$, then the next vector x_{i+1} contains frames from $i+1$ to $i+l$). This way, each vector represents a $f \times l$ patch of the spectrogram, where f is the number of frequency bins. Overall, a spectrogram of t time-frames with f bins results in a sequence of $t - l + 1$ vectors with fl components each.

For each vector in the resulting sequence, a sparse representation is calculated. We aggregate the information from the sequence of sparse vectors via pooling, two methods of pooling are considered: max pooling and average pooling.

In max pooling, a sequence of n-dimensional vectors is aggregated to a single n-dimensional vector by choosing the maximum value of i-th dimension among all vectors in the sequence as the i-th component of the resulting vector. In average pooling, we simply sum vectors in the sequence and divide the result by the number of vectors. For both pooling methods, the length of resulting vector representing the music file will be equal to the size of sparse representation vector. It is possible to use one of the methods over the entire file, or to mix them by first splitting the sequence of vectors into fragments of length m, use one type of pooling over the fragment, resulting in a sequence that is m times shorter, and then aggregate that sequence using the second type of pooling. The process is shown in Fig. 1.

The intuitive rationale behind using pooling for a sequence of sparse vectors is that every dimension in sparse representation corresponds to a particular pattern which can be found in the data used in the process of dictionary building or learning the encoding neural network. With this interpretation, we should understand max pooling as a method to find out whether a particular pattern appeared in a given part of a song at all. If maximal value of i-th element over multiple vectors in a sequence is low, there was no vector that could be reconstructed using i-th dictionary element or a vector that would result in a strong activation of i-th neuron. Similarly, average pooling gives the information of a particular dictionary element (or a pattern detected by the neural network) appearing consistently over the course of the entire song.

One useful property of max pooling is that it eliminates the drawback of autoencoders possibly creating non-sparse representation of data in the strict sense. The notion of 'sparse' vectors in which most neurons are 'inactive', while not precise, is not a problem here because the low activation values of 'inactive' neurons are mostly lost in the process of pooling regardless of whether or not they equal 0.

VI. Experiments on Emotify Dataset

The main goal of our experiments was to compare the performance of different sparse coding methods in creating features for emotion regression task (subsection VI-B). Additionally we tested the influence of the pooling window size on the result (subsection VI-C. Our third experiment shows the spectrogram patterns recognized by sparse coding methods (subsection VI-D. At the beginning we present what is common across all performed experiments. The used tools and

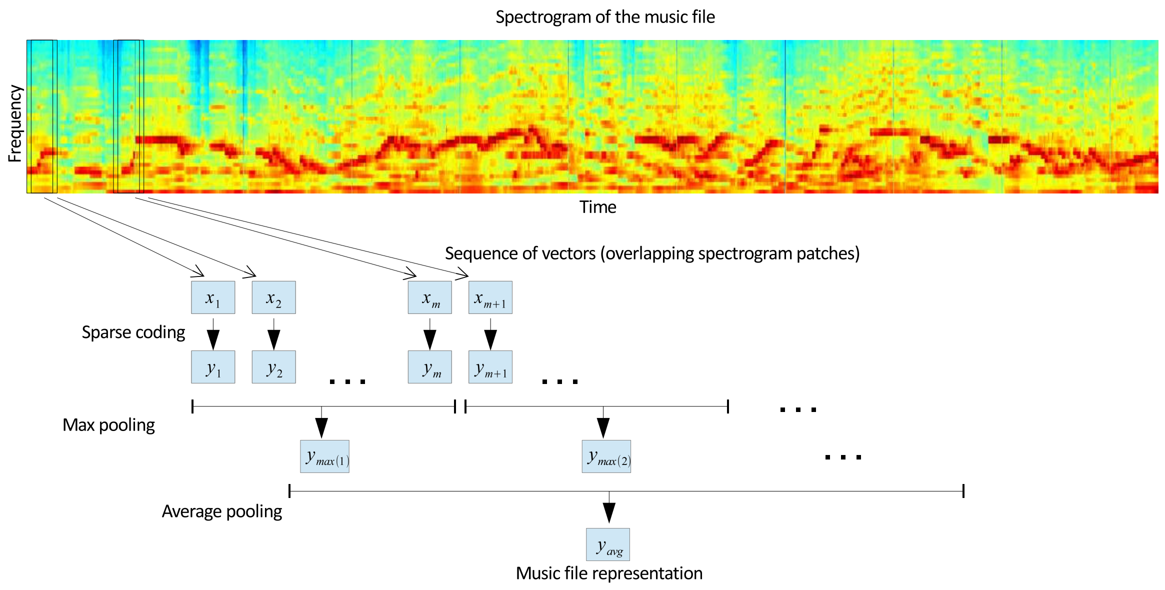
Spectrogram of the music file
Frequency
Time
Sequence of vectors (overlapping spectrogram patches)
x_1 x_2 x_m x_{m+1}
Sparse coding
y_1 y_2 y_m y_{m+1}
Max pooling
$y_{max(1)}$ $y_{max(2)}$
Average pooling
y_{avg}
Music file representation

Fig. 1. Representation of a music file by pooling sparse vectors

the learning of sparse representation as well as the regression algorithm is presented in subsection VI-A.

A. Common conditions of the experiments

All performed experiments were performed using MATLAB/Octave toolboxes: MIR Toolbox [31] for spectrogram extraction from music files, Sparse Representation Toolbox [32] for the L1 regularized least squares implementation and DeeBNet [33] for the implementation of autoencoder and sparse RBM.

Sparse Representation Learning
Representation of music files we used in our experiments was based on spectrograms. Spectrogram of each file in the dataset was extracted using 50 ms frames with 25 ms overlap, using mel scale with $f = 40$ bins for frequency axis. The resulting values were logarithmized. Spectrogram patches were $l = 10$ frames long. This means vectors of length $fl = 400$ were used as an input for sparse coding algorithms.

We report the results for sparse representation size of $k = 200$ to emphasize tested methods' capabilities to build compact representations of musical files. In comparison, the best-performing feature set in previous research on the Emotify dataset consisted of 6535 features [18]. Testing sparse representation sizes 200, 500 and 1000 we found that increasing the vector size beyond 200 does not improve the performance significantly. For efficiency, both dictionary building and neural network training were performed on a randomly selected set of 40000 spectrogram patches from the whole dataset.

For kernelized version of L1-regularized least squares algorithm, we use radial basis function kernel defined as:

$$K(x,y) = \exp(-\gamma\|x-y\|^2) \tag{9}$$

Scaling parameter λ, which governs the weight of sparsity constraints relatively to reconstruction accuracy, is present in all methods except K-means. For this parameter we considered values of $\{0.01, 0.1, 1, 10, 100\}$. For parameter ρ, the desired average activation of neurons in sparse autoencoder and sparse RBM, values $\{0.01, 0.05, 0.1, 0.2, 0.5\}$ were considered. These parameters were optimized using grid search, and we report the best results.

Regression algorithm
We approached music emotion recognition as a regression problem: the input was the representation of the file calculated by pooling sparse vectors, i.e. a vector with 200 elements, and the value we were attempting to predict was a real number within [0,1] range, corresponding to the level of agreement that the music file evokes a certain emotion in the listener, as described in section III. A separate Support Vector Regression (SVR) [34] model was trained for each emotion. SVR is a model in which data is fit by a hyperplane, however, due to the possibility of using kernel trick one can accommodate for non-linear data. In our experiments we found that radial basis function kernel, defined as in (9), performs the best.

B. Experiment 1: Performance of different sparse coding methods

Obtained results are collected in Table 1. In order to compare them with [18], which is the only MER paper that reported results of research on Emotify dataset so far, we use the same performance measure, namely Pearson's correlation coefficient R. We compare proposed sparse coding methods to the results achieved using a feature set consisting of features available in MIRToolbox and harmonic features based on chord detection and interval detection.

TABLE I
SPARSE CODING METHODS COMPARED WITH A HIGHER-LEVEL FEATURE SET (PERFORMANCE MEASURED BY PEARSON'S R)

	K-means	**L1 Least Squares**	**Kernel L1LS**	**Autoencoder**	**Sparse RBM**	**MIRtoolbox+Harmonic [18]**
Amazement	0.27	0.20	0.18	**0.29**	0.21	0.16
Solemnity	0.41	0.43	0.36	**0.50**	0.48	0.43
Tenderness	0.30	0.46	0.38	0.54	0.49	**0.57**
Nostalgia	0.47	0.40	0.37	**0.50**	0.40	0.45
Calmness	0.47	0.47	0.45	**0.56**	0.53	0.50
Power	0.47	0.46	0.44	0.53	0.50	**0.56**
Joyful activation	0.48	0.50	0.50	0.53	0.54	**0.66**
Tension	0.32	0.47	0.26	**0.48**	0.43	0.46
Sadness	0.30	0.32	0.27	0.33	0.27	**0.42**
Average	0.39	0.43	0.37	**0.47**	0.43	**0.47**

For pooling, we first use max pooling over fragments of music file corresponding to 100 spectrogram patches (around 3 seconds), and then average pooling over the resulting sequence. Results were measured using 10-fold cross-validation.

It can be seen that on average, a sparse coding based approach can achieve performance comparable to that of hand-crafted features. However, autoencoder-based music representation outperforms hand-crafted features features in modeling highly subjective emotions, such as amazement and solemnity. At the same time, the results are significantly worse for "simple" emotions highly correlated with occurrence of major and minor chords, i.e. joyful activation and sadness.

Out of all examined methods, only autoencoders achieved this level of performance. The addition of radial basis function kernel to the L1 least squares not only did not improve the results, but worsened them. Sparse RBM performs better than most approaches, but does not achieve the performance of a sparse autoencoder. Interestingly, K-means based approach, equivalent to the simple bag of words model, achieves performance comparable to autoencoder. However, it does so only for specific emotions, while being significantly worse for others.

Under-performing in terms of recognizing joyful activation and sadness can be interpreted as a sign of difficulty in learning harmonic features. While an unsupervised dictionary learning method can recognize a pattern corresponding to a particular chord, it may be hard to generalize them to the concept of major and minor chords, and further to key detection for the entire song.

C. Experiment 2: Influence of pooling window size on the results

Using previous parameters and choosing autoencoder as the best performing method of sparse coding, we performed a second experiments with the goal of estimating the influence of selected pooling window size on the prediction of specific emotions. Usually, in music information retrieval, model parameters such as window sizes for any type of operation are constant and tuned for best overall performance. However, with a specific set of differing emotions, it is interesting to see how recognition of each emotion is affected by the time-frame considered.

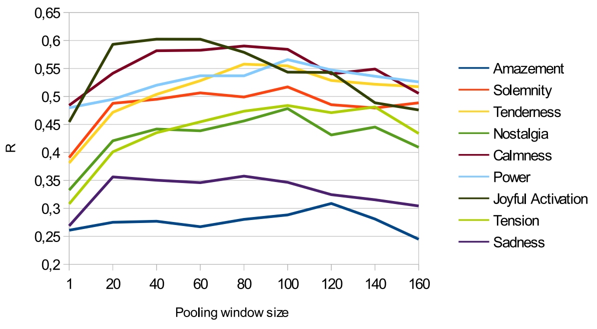

Fig. 2. Performance depending on pooling window size

As in the experiment 1, we use max pooling over small windows first, and average pooling over the resulting sequence of vectors afterwards. We consider pooling window sizes of $\{1, 20, 40, 60, 80, 100, 120, 140, 160\}$. Note that window size equal to one is equivalent to average pooling over the entire file, which is commonly used in dictionary learning. Similarly to the main experiment, results are measured using 10-fold cross-validation.

Fig. 2 shows the changes in performance for different window sizes of max pooling. It can be seen that peaks in performance appear at different window sizes for different emotions. For detection of joyful activation and sadness small window sizes seem optimal, while detection of tension can be improved with sizes close to 140 vectors.

D. Experiment 3: Encoding patterns

Since our sparse representations are built from spectrogram patches and spectrogram is a visual representation of how sound evolves, it is possible to visualize spectrogram patterns which sparse coding methods recognize. In case of autoencoder neural network, a response of a hidden layer neuron depends on the vector of input-hidden weights associated with that particular neuron. For each neuron in the hidden layer, there are 400 weights, each corresponding to one input neuron. Since a vector given as an input to the neural network represents a 40×10 spectrogram patch, we use the same 40×10 dimensions to visualize the input weights. We want to see if the response patterns are possible to interpret by a human observer and relate to concepts existing in music. For visualization of encoding patterns, we learned an autoencoder network with the best-performing set of parameters found in previous experiments: $\lambda = 10$, $\rho = 0.1$.

Fig. 3 shows sample response patterns of single neurons in the encoding layer, with the highest weight values coloured red and lowest coloured blue. Neuron should respond with a high activation to spectrogram patches similar to these patterns.

We found certain types of response patterns appearing consistently in unsupervised training of autoencoders. In Fig. 3, leftmost pattern shows high weights in distinct narrow frequency bands, responding in high activation when the amplitude is high in these bands. Patterns like this respond well to specific music intervals, which makes them useful in extracting harmonic features. A combination of them could approximate chord detection. Second leftmost pattern shows an example of high weights in a wider frequency band, in this case responding to high amplitude in low frequency bins. Neurons responding to patterns like this for different wide bands can enable detection of overall amount high, mid and low tones, which in turn could be used e.g. in recognizing songs with a heavy bass line. Third pattern shows small weights for inputs corresponding to initial frames and high weights for inputs corresponding to latter frames in the low frequency band, indicating the neuron will respond with high activation to an onset of a musical phrase. Rightmost pattern can detect increases in amplitude separated by a specific interval of time, which gives the possibility of beat frequency detection.

We can see that through unsupervised training, autoencoders could learn features that are interpretable using concepts related to high-level features of music.

VII. Conclusions and Future Work

We applied a machine learning approach based on pooling of sparse vectors built from spectrograms of sound files to Emotify, first publicly available music emotion recognition dataset annotated with Geneva Emotional Music Scale categories. We compared 5 sparse coding methods and measured their performance in the task of predicting a community consensus concerning emotions induced by a music piece. We found out that sparse autoencoders significantly outperform other approaches.

The results show that a sparse coding approach based on autoencoders can achieve satisfying performance in recognizing certain very subjective emotions hard to detect using higher level features. On the other hand even the best sparse coding method applied to this problem cannot outperform harmonic features in recognizing joyful activation and sadness. However, the results show that performance is significantly affected

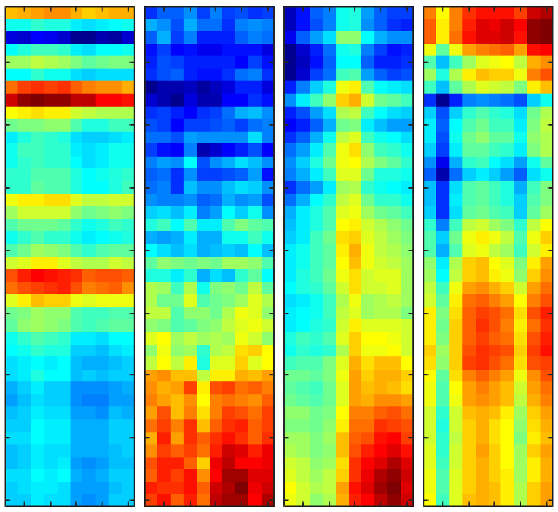

Fig. 3. Encoding patterns: input weights of four sample encoding layer neurons

by the choice of the size of pooling window and it appears that different window sizes are optimal for detecting different emotions. These results suggest that analysing sound using fixed and constant time scales is a significant limiting factor in predicting music-induced emotions.

The main limitation of our research was the dataset, which unfortunately is the only one so far using the GEMS system for emotion classification. Generalizations based on our results should be confirmed on other datasets, possibly including more varied music types.

Future research in the area of induced emotion modelling can focus on a number subjects. First, a thorough examination of properties of the sparse autoencoder in the context of music emotion recognition can be useful. Second possible area of research is developing methods that can simultaneously consider multiple time-scales since there is no optimal selection of time windows for pooling, and maximizing average prediction performance does not guarantee maximizing performance for every emotion. Finally, building new datasets annotated with domain-specific emotional models such as GEMS should be a subject of interest.

References

[1] Y. E. Kim, E. M. Schmidt, R. Migneco, B. G. Morton, P. Richardson, J. Scott, J. A. Speck, and D. Turnbul, "Music emotion recognition: a state of the art review," *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, 2010.

[2] K. R. Scherer, M. Zentner, "Emotion effects of music: Production rules", in: *Music and emotion: Theory and research*, pp. 361-392, Oxford University Press, 2001.

[3] Qin, Zengchang et all, "A Bag-of-Tones Model with MFCC Features for Musical Genre Classification," in: Advanced Data Mining and Applications: 9th International Conference Proceedings, Part I. ADMA 2013, p. 564–575, Springer Berlin Heidelberg, 2013.

[4] E. J. Humphrey, J. P. Bello, and Y. LeCun, "Moving beyond feature design: Deep architectures and automatic feature learning in music informatics," in *Proceedings of the 13th International Conference on Music Information Retrieval (ISMIR)*, 2012.

[5] M. Henaff, K. Jarrett, K. Kavukcuoglu, and Y. LeCun, "Unsupervised learning of sparse features for scalable audio classification," in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[6] S. Sigtia and S. Dixon, "Improved music feature learning with deep neural networks," in *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing* (ICASSP), 2014.

[7] Y. Vaizman, B. McFee, and G. Lanckriet, "Codebook based audio feature representation for music information retrieval," *IEEE Transactions on Acoustics, Speech and Signal Processing*, 2014.

[8] J. Nam, J. Herrera, M. Slaney, and J. Smith, "Learning sparse feature representations for music annotation and retrieval," in *Proc. ISMIR,* 2012.

[9] T. Li and M. Ogihara, "Detecting emotion in music," in *Proc.of the Intl. Conf. on Music Information Retrieval*, Baltimore, MD, October 2003.

[10] J. Skowronek, M. McKinney, and S. van de Par, "A demonstrator for automatic music mood estimation," in *Proc. Intl. Conf. on Music Information Retrieval*, Vienna, Austria, 2007.

[11] C. Laurier, O. Lartillot, T. Eerola, and P. Toiviainen: "Exploring Relationships between Audio Features and Emotion in Music," *Conference of European Society for the Cognitive Sciences of Music*, 2009.

[12] Y. H. Yang, Y. C. Lin, Y. F. Su, and H. H. Chen, "A Regression Approach to Music Emotion Recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 16, No. 2, pp. 448-457, 2008.

[13] P. Ekman, "An argument for basic emotions," *Cognition Emotion* vol. 6, no. 3, pp. 169-200, 2001.

[14] K. Hevner, "Experimental studies of the elements of expression in music," *American Journal of Psychology*, vol. 48, pp. 248 -268, 1936.

[15] Y. E. Kim, E. M. Schmidt, R. Migneco, B. G. Morton, P. Richardson, J. Scott, J. A. Speck, and D. Turnbul, "Music emotion recognition: a state of the art review," *11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, 2010.

[16] U. Schimmack and R. Reisenzein, "Experiencing activation: energetic arousal and tense arousal are not mixtures of valence and activation," *Emotion*, vol. 2, no. 4, p. 412, 2002.

[17] M. Zentner, D. Grandjean, and K. R. Scherer: "Emotions evoked by the sound of music: characterization, classification, and measurement," *Emotion*, vol. 8, no. 4, pp. 494-521, 2008.

[18] A. Aljanaki, F. Wiering, and R. Veltkamp, "Computational modeling of induced emotion using GEMS," *Proceedings of the 15th Conference of the International Society for Music Information Retrieval (ISMIR 2014)*, pp. 373-378, 2014.

[19] G. E. Hinton, "A practical guide to training restricted Boltzmann machines," Tech. Rep. UTML TR 2010-003, Dept. Comput. Sci., Univ. Toronto, 2010.

[20] N.J. Nalini, S. Palanivel, "Emotion Recognition in Music Signal using AANN and SVM," *International Journal of Computer Applications* vol. 77, no.2, 2013.

[21] N. Glazyrin , "Mid-level features for audio chord recognition using a deep neural network," *Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko-Matematicheskie Nauki*, vol. 155, no. 4, pp. 109–117, 2013.

[22] A. Aljanaki, D. Bountouridis, J.A. Burgoyne, J. van Balen, F. Wiering, H. Honing, and R. C. Veltkamp, "Designing Games with a Purpose for Data Collection in Music Research. Emotify and Hooked: Two Case Studies," *Proceedings of Games and Learning Alliance Conference,* 2013.

[23] B. Logan, "Mel frequency cepstral coefficients for music modeling," in *Proc. of the Intl. Symposium on Music Information Retrieval*, Plymouth, MA, 2000.

[24] J. B. MacQueen, "Some Methods for classification and Analysis of Multivariate Observations," *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, 1967.

[25] F. Bach, R. Jenatton, J. Mairal, G. Obozinski, "Optimization with Sparsity-Inducing Penalties," *Foundations and Trends in Machine Learning* vol. 4, no. 1, 2012.

[26] S. Gao, I. W. H. Tsang, L. T. Chia, "Sparse representation with kernels," in *IEEE Transactions on Image Processing*, vol. 22, no. 2, pp. 423-434, 2012.

[27] Y. Bengio, "Learning Deep Architectures for AI", *Foundations and Trends in Machine Learning,* vol. 2, no. 1, 2009.

[28] D. E. Rumelhart, G. E. Hinton, Williams, Ronald J. . "Learning representations by back-propagating errors", *Nature*, vol. 323, pp. 533-536, 1986.

[29] Q. V. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A. Y. Ng, "On optimization methods for deep learning," in *Proc. 28th Int. Conf. Machine Learning,* pp. 265–272, 2011.

[30] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computing*, vol. 14, pp. 1771-1800, 2002.

[31] Olivier Lartillot, Petri Toiviainen, "A Matlab Toolbox for Musical Feature Extraction From Audio," *International Conference on Digital Audio Effects,* Bordeaux, 2007.

[32] Y. Li "Sparse representation for high-dimensional data analysis," in *Sparse Machine Learning Models in Bioinformatics,* PhD Thesis, School of Computer Science, University of Windsor, Canada, 2013.

[33] M. A. Keyvanrad and M. M. Homayounpour, "A brief survey on deep belief networks and introducing a new object oriented toolbox (DeeBNet)," arXiv:1408.3264 [cs], Aug. 2014.

[34] A. J. Smola, B. Scholkopf, "A tutorial on support vector regression," *Statistics and Computing,* vol. 14, no. 3 , pp 199-222, 2004.