Hindawi
Complexity
Volume 2018, Article ID 1935938, 10 pages
https://doi.org/10.1155/2018/1935938

WILEY | Hindawi

Research Article

Similarity-Based Summarization of Music Files for Support Vector Machines

Jan Jakubik and Halina Kwaśnicka

Department of Computational Intelligence, Faculty of Computer Science and Management, Wrocław University of Science and Technology, Wrocław, Poland

Correspondence should be addressed to Jan Jakubik; jan.jakubik@pwr.edu.pl

Received 19 April 2018; Accepted 4 July 2018; Published 1 August 2018

Academic Editor: Piotr Jędrzejowicz

Automatic retrieval of music information is an active area of research in which problems such as automatically assigning genres or descriptors of emotional content to music emerge. Recent advancements in the area rely on the use of deep learning, which allows researchers to operate on a low-level description of the music. Deep neural network architectures can learn to build feature representations that summarize music files from data itself, rather than expert knowledge. In this paper, a novel approach to applying feature learning in combination with support vector machines to musical data is presented. A spectrogram of the music file, which is too complex to be processed by SVM, is first reduced to a compact representation by a recurrent neural network. An adjustment to loss function of the network is proposed so that the network learns to build a representation space that replicates a certain notion of similarity between annotations, rather than to explicitly make predictions. We evaluate the approach on five datasets, focusing on emotion recognition and complementing it with genre classification. In experiments, the proposed loss function adjustment is shown to improve results in classification and regression tasks, but only when the learned similarity notion corresponds to a kernel function employed within the SVM. These results suggest that adjusting deep learning methods to build data representations that target a specific classifier or regressor can open up new perspectives for the use of standard machine learning methods in music domain.

1. Introduction

Recently, in our digital world, there are huge resources of data, images, video, and music. Advanced methods of automatic processing of music resources remain in the sphere of interest of many researchers. The goal is to facilitate music information retrieval (MIR) in a personalized way for the needs of an individual user. Despite the involvement of researchers and use of state-of-the-art methods, such as deep learning, there is a lack of advanced search engines, especially able to take into account users' personal preferences. Observed quick increase in the size of music collections on the Internet resulted in the emergence of two challenges. First is the need for automatic organizing of music collections, and the second is how to automatically recommend new songs to a particular user, taking into account the user's listening habit [1]. To recommend a song according to user's expectations, it is beneficial to automatically recognize the emotions that a song induces to the user and the genre to which a song belongs.

Music, similarly to a picture, is very emotionally expressive. In developing system for music indexing and recommendation, it is necessary to consider emotional characteristics of music [2]. Identifying the emotion induced by music automatically is not yet solved and the problem remains a significant challenge. The relationship between some basic features as timbre, harmony, or lyrics and emotions they can induce is complex [3]. Another problem is a high degree of subjectivity of emotions induced by music. Even if we take into account the same listener, then the emotions induced by a given piece of music may depend on their mood, fatigue, and other factors. All of the above makes the automatic recognition of emotions (by classification or regression) a difficult task.

In emotion recognition, there are categorical [4] and continuous space [5] models of emotion; both are research topics [6, 7]. The most popular model is two-dimensional continuous valence-arousal scale. Positive and negative emotions are indicated on one coordinate axis, and arousal separates low activation from high on the second. This model of emotions is derived from research concerning emotions in general. Authors of [8] consider emotion recognition as a regression separately for arousal and valence. Other types of emotions are considered in Geneva Emotional Music Scale (GEMS) [9]. Categories defined in GEMS are domain-specific. They are the result of surveys in which participants were asked to describe emotion induced by listened music. Emotions in GEMS are organized in three levels: the higher level contains generic emotion groups; the middle level consists of nine categories: wonder, transcendence, tenderness, nostalgia, calmness, power, joy, tension, and sadness; and the lowest contains specific nouns.

Another research topic in MIR area is the problem of automatic classification of music pieces taking into account genre [10]. In music analysis, genre represents a music style. Members of the same style (genre) share similar characteristics such as tempo, rhythm patterns, and types of instruments and thus can be distinguished from other types of music.

As music data is extremely complex, the key issue when handling it in machine learning systems becomes summarization of them in a form that a classifier can process. While research datasets typically employed in MIR studies are not large in terms of file count, the complexity and variety within each individual file are significant. For both genre and emotion recognition, the use of machine learning methods is largely reliant on the appropriate selection of features that describe the music samples. In general, automatic music analysis such as music classification (or regression when we deal with emotion recognition) encompasses two steps: feature extraction and the classification (regression). Both are difficult and strongly influence the final result. Early works used manually defined set of features based on expert domain knowledge. Many researchers have studied the relationship between emotion and different features that describe music [5]. In [11], the authors added harmonic features to a set of popular music features to the predicting community consensus task with GEMS categories. They show that adding harmonic features improves the accuracy of recognition.

The authors of [12] proposed the use of feature learning on low-level representations instead of defining a proper set of features manually. Codebook methods have been shown to learn useful features even in shallow architectures [13–15]. The use of simple autoencoder neural network to learn features on a spectrogram for predicting community consensus task with GEMS categories gives comparable results as traditional machine learning with the use of a manually well-chosen set of features [16]. Deep learning improves these results further, resulting in state-of-the-art performance. Convolutional recurrent neural networks, working on a low-level representation of sounds, have been used for learning features that would be useful in classification task [17, 18]. While deep learning in itself performs very well, it creates new opportunities for the use of older machine learning methods. The features can be taken from the selected level of deep network and used as an input to a support vector machine (SVM), or a regression method such as SVR, or any other classifier [19].

In our research, we are interested in the possibility of improving the usefulness of traditional machine learning methods, in particular, SVM, when combined with deep learning as a feature extractor. For training a deep neural network for classification, typically the softmax activation function for prediction and minimization of cross entropy loss is employed. Effectively, the network is trained to maximize the performance of its final layer, which works as a linear classifier on features from the previous layers. However, one of the biggest advantages of SVM among standard machine learning methods is its performance on nonlinear problems. It is largely reliant on the so-called kernel trick—replacing the inner product in the solved optimization problem with kernel functions, which can be understood as similarity measures. Given that a neural network can be trained to minimize any loss differentiable with respect to the network's weight matrices, it may be possible to adjust it so that it produces features specifically fit for a kernel SVM, rather than a linear classifier. Knowing the basic principle of kernel trick, we attempt to train the network to replicate certain notion of similarity between annotations that describe genres or emotions of the music pieces, within representation space that is the output of a neural network. The goal of this study is to test whether the proposed change in the approach to training the feature extracting network will yield performance improvements over simply using an NN for both feature learning and classification or regression, as well as SVM deployed on features extracted from a NN learned with a standard loss function.

Our approach is similar to the one presented in [20], where the author replaces the softmax layer of an NN by linear SVM. However, the approach presented by Tang is concerned with the integration of linear SVM within the network. In contrast, we treat SVM as a classifier separate from the feature learning process, assuming the feature learning takes place first, and then the classifier is trained on features extracted by the network. This is in line with the growing trend of transfer learning, which seeks to reuse the complex architectures trained on large datasets, for multiple problems. A feature extracting network could be easily reused on other similar tasks while only retraining the classifier SVM, similarly to [21].

We consider tasks of classification and regression on five different datasets. Focusing on emotion, we use three music mood recognition datasets, one for classification and two for regression. We complement these with two classification datasets, one for genre recognition and one for dance style recognition. The paper is organized into three sections: "Introduction," "Materials and Methods," and "Results and Discussion." The second section contains all theoretical background, dataset descriptions, and other information required to replicate the study, while in the third, we present and discuss the obtained results.

2. Materials and Methods

The goal of our research is to evaluate the possibility of using recurrent neural networks as a feature learner while changing its loss function to one based on pairwise similarity rather than one explicitly predicting annotations within the network. We hypothesize this approach will better fit an SVM-based classifier or regressor. This section contains a description of neural network architectures employed in the study and the datasets on which we performed our experiments. Conditions of the experiments, such as hyperparameters of the algorithms, are also described. We refrain from explaining SVM in detail, as our contribution does not develop the method itself.

2.1. Gated Recurrent Neural Networks. Recurrent neural networks (RNN) are useful for modelling time series [22]. A basic recurrent layer is defined by

$$h_t = \sigma(\mathbf{W}\mathrm{x}_t + \mathbf{U}h_{t-1} + \mathbf{b}), \tag{1}$$

where σ is an activation function, which can be logistic sigmoid function (σ_{sig}) or hyperbolic tangent activation ($\sigma_{\tanh}$); $\mathbf{W}$ and $\mathbf{U}$ are matrices of weight; and $\mathbf{b}$ is the bias vector. x_t is a current input, in a series of l input vectors, $(x_1, x_2, \dots, x_l)$. Matrices $\mathbf{W}$ and $\mathbf{U}$ and the bias vector $\mathbf{b}$ are learned using backpropagation algorithm.

As the more complex models, with the use of gating mechanisms, have been applied to natural language processing with success, they became a common research subject within the deep learning area. In these, a special "unit" replaces a recurrent layer. It consists of multiple interconnected layers. Outputs can be multiplied or added element-wise. When element-wise multiplication of any output with an output of a log-sigmoid layer is applied, a "gating" mechanism is created. The log-sigmoid layer is a kind of gate that decides if the output passes (multiplication by 1) or not (multiplication by 0). Long short-term memory (LSTM) [23] network is the most popular model that uses gating. LSTM is defined by

$$\begin{aligned}
r_t &= \sigma_{\mathrm{sig}}(\mathbf{W}_r x_t + \mathbf{U}_r h_{t-1} + \mathbf{b}_r),\\
i_t &= \sigma_{\mathrm{sig}}(\mathbf{W}_i x_t + \mathbf{U}_i h_{t-1} + \mathbf{b}_i),\\
o_t &= \sigma_{\mathrm{sig}}(\mathbf{W}_o x_t + \mathbf{U}_o h_{t-1} + \mathbf{b}_o),\\
c_t &= r_t \circ c_{t-1} + i_t \circ \sigma_{\tanh}(\mathbf{W}_o x_t + \mathbf{U}_o h_{t-1} + \mathbf{b}_o),\\
h_t &= o_t \circ \sigma_{\tanh}(c_t),
\end{aligned} \tag{2}$$

where r_t, i_t, and o_t are the outputs of gates (standard log-sigmoid recurrent layers); $\mathbf{W}_r$, $\mathbf{U}_r$, $\mathbf{W}_i$, $\mathbf{U}_i$, $\mathbf{W}_o$, and $\mathbf{U}_o$ are weight matrices; $\mathbf{b}_r$, $\mathbf{b}_i$, and $\mathbf{b}_o$ are bias vectors; and $\circ$ denotes element-wise multiplication. c_t is a cell memory stat; it is calculated using the two weight matrices $\mathbf{W}_c$ and $\mathbf{U}_c$ and a bias vector $\mathbf{b}_c$.

The authors of [24] present a simplified version of gated model that gives results similar to LSTM. Gated recurrent unit (GRU) reduces the internal complexity of a unit; it is defined by

$$\begin{aligned}
z_t &= \sigma_{\mathrm{sig}}(\mathbf{W}_z x_t + \mathbf{U}_z h_{t-1} + b_z),\\
r_t &= \sigma_{\mathrm{sig}}(\mathbf{W}_r x_t + \mathbf{U}_r h_{t-1} + \mathbf{b}_r),\\
c_t &= r_t \circ h_{t-1},\\
h_t &= z_t \circ h_{t-1} + (1 - z_t) \circ \sigma_{\tanh}(\mathbf{W}_h x_t + \mathbf{U}_h c_t + b_h).
\end{aligned} \tag{3}$$

In GRU, the memory state is not separated from the output. The output depends only on the current input and the value of the previous output. GRU uses two gates z_t and r_t. As c_t represents the previous output after gating, there is no need to store it between timesteps. The numbers of weight matrices and bias vectors are reduced in GRU to six matrices ($\mathbf{W}_z$, $\mathbf{U}_z$, $\mathbf{W}_r$, $\mathbf{U}_r$, $\mathbf{W}_h$, and $\mathbf{U}_h$) and three bias vectors ($\mathbf{b}_r$, $\mathbf{b}_i$, and $\mathbf{b}_o$). Chung et al. compared GRU and LSTM in [25]. Both networks perform similarly and better than standard recurrent neural networks. The advantage of GRU lies in its simplicity, comparing to LSTM; therefore, we prefer GRU networks in our studies.

2.2. Similarity-Based Loss for a Neural Network. A GRU network produces a sequence of feature vectors in its final layer. For a sequence of n output vectors $(h_1, h_2, \dots, h_n)$, that is, the result of input $(x_1, x_2, \dots, x_n)$, we can calculate the average to obtain a feature vector f describing the whole music piece:

$$f(x_1, x_2, \dots, x_n) = \frac{1}{n}\sum_{t=1}^{n} h_t, \tag{4}$$

where the sequence $(h_1, h_2, \dots, h_n)$ is calculated according to (3). The standard approach for training recurrent networks for sequence classification is to use this vector as an input to a final nonrecurrent layer. A loss function is then calculated using mean square error or (after applying softmax function over outputs) cross entropy. We seek an adjustment to loss function that would take into account the properties of SVM as nonlinear classifiers and the fact we can simply ignore the need of a nonrecurrent layer if we use the network as a dedicated feature learner.

A particularly well-known way to improve the performance of SVMs is to use the so-called kernel trick. Assume an optimization problem that is posed in such a way that it does not require access to a full data matrix $\mathbf{D}$, but rather, a product of the matrix and its transpose $\mathbf{D}\mathbf{D}^{\mathrm{T}}$. Linear SVM is an example of such problem. Then, we can replace $\mathbf{D}\mathbf{D}^{\mathrm{T}}$ with a matrix $\mathbf{K}(\mathbf{D}, \mathbf{D})$, built using a real-valued kernel function:

$$K(X, Y)_{ab} = k(x_a, y_b), \tag{5}$$

whereby $K_{a,b}$ denote an element of matrix $\mathbf{K}$ in ath row, bth column, while x_a denotes the ath row of matrix $\mathbf{X}$. In other words, the kernel function k replaces the inner product during optimization. If there is a mapping ϕ such that

$$k(x, y) = \phi(x)^T \phi(y), \tag{6}$$

we can say that the problem is instead being solved in an implicit feature space, where the coordinates of the classified samples x and y are $\phi(x)$ and $\phi(y)$, respectively. In this space,

certain classification problems which were not linearly separable in the original feature space may become linearly separable. Similarly, for regression problems in which linear regression produced a bad fit, regression in implicit feature space often improves results. The advantage of kernel trick is that it allows operating within the implicit feature space without actually calculating $\phi(x)$ and $\phi(y)$.

Kernel functions typically employed in SVM training can be understood as measures of similarity. Our intuition for the feature learning NN is, therefore, to attempt to replicate certain similarity relations between the annotations in the learned feature space. We can stack feature vectors calculated according to (4) for different files in the dataset as rows of a feature vector matrix **F**. Similarly, known annotation vectors for these files form an annotation matrix **A**. For regression problem in a multidimensional space, these annotations consist of all regressed values. For example, for a music piece annotated with two values regarding its position on valence-arousal plane, a vector of two real values is the annotation. For classification, we can consider one-hot encoding of classes. We can define a similarity-based loss function as

$$L(X) = \|\mathbf{K}(\mathrm{F}, \mathbf{F}) - \mathbf{K}(\mathbf{A}, \mathbf{A})\|, \tag{7}$$

where K can be built using an arbitrary notion of similarity $k(x, y)$, by analogy to kernel SVM, as in (5), and $\| \dots \|$ denotes Frobenius norm. For batch learning, which is currently the standard procedure for learning neural networks due to performance considerations, the matrices $\mathbf{K}(\mathbf{F}, \mathbf{F})$ and $\mathbf{K}(\mathbf{A}, \mathbf{A})$ can be calculated over batches instead of full dataset. We described this approach in less general terms in [19] as semantic embedding, borrowing the idea from the domain of text processing [26]. Semantic embedding in texts seeks to learn similarity between documents using pairs of similar and dissimilar files and could be considered a special case of the described idea (with cosine similarity as the k function and $\mathbf{K}(\mathbf{A}, \mathbf{A})$ being built as a matrix of ones and zeroes from known relation of similarity, rather than calculated from annotations).

2.3. Measures of Similarity between Vectors. To define a similarity-based loss, we need to define a similarity function that will be used. For the purpose of this study, we focus on three similarity measures:

(i) *Cosine*: the similarity notion that we used in the earlier paper [19], where we first tackled learning similarity. It was previously used in the approach to learning similarity between documents called semantic embedding.

(ii) *Radial basis function (RBF) kernel*: one of the popular kernels often employed in support vector machines and the one we use in the SVM classifier or regressor deployed on learned features.

(iii) *Polynomial kernel*: the other popular kernel employed in support vector machines which we use for comparison. We need this comparison to establish whether the performance gains are related to fitting specific similarity notion to the kernel employed in SVM or simply rewriting loss function to use similarity yields benefits over a loss function that tries to predict labels directly

Cosine similarity is a simple measure that normalizes both compared vectors, therefore ignoring their norm and only focusing on the direction (i.e., for a vector **x**, $\cos(x, 2x) = 1$). The function is defined as

$$k_{\cos}(x, y) = \frac{x^T y}{\|x\| \|y\|}. \tag{8}$$

Cosine similarity is bound between 0 and 1 regardless of space dimensionality, which may be a useful property for our purposes as annotation space and learned feature space could have largely varying dimensionalities. Radial basis function kernel is defined as

$$k_{\mathrm{rbf}}(x, y) = e^{-\gamma \|x-y\|^2}. \tag{9}$$

The exponent guarantees that the value is in the $(0, 1]$ interval and the similarity between two vectors never equals 0. In practice, the lower bound of this measure will be affected by the maximal distance between vectors which will exist in real datasets. For example, for annotation space of n dimensions, if we assume all labels can range from 0 to 1 (as in the Emotify dataset), the distance between two annotations can be at the most square root of n. The lower bound for similarity is therefore $e^{-\gamma n}$.

Polynomial kernel is defined as

$$k_{\mathrm{pol}}(x, y) = \left(x^T y + b\right)^p. \tag{10}$$

The polynomial kernel is not bound to a particular interval (although for even p result is always nonnegative), and the result is greater when comparing vectors with larger norms. Polynomial kernel properties are not theoretically fit for our task since dimensionality would largely affect the similarity score between vectors. However, in preliminary studies, we found it performed surprisingly well in classification tasks even despite the fact that SVM was using an RBF kernel. Therefore, we include it in the study as a possible RBF kernel alternative.

2.4. Datasets. We performed our experiments on five datasets, two for regression and three for classification. These datasets represent three distinctive label types, with focus on emotion recognition. Links to all datasets are provided at the end of the article, in the "Data Availability" statement. A short summary is presented in Table 1.

We chose both datasets that were previously tested in [19] and three complementary datasets. Complementary data represents an important form of emotion regression (predicting the values of valence and arousal) and two music classification tasks not concerning emotion. We found it important to extend our research to V-A emotion recognition, as it is the most common form of annotating emotion in the existing literature.

Table 1: Summary of the datasets.

Dataset	Label type	Task	Labels	Files
Lastfm	Emotion	Classification	4	2906
Emotify	Emotion	Regression	9	400
Songs	Emotion	Regression	2	744
GTZAN	Genre	Classification	10	1000
Ballroom	Dance style	Classification	9	754

The Lastfm dataset [27] is the largest one we test. It contains more than 2000 files annotated with labels inferred from user-generated tags on the music-centric social network https://www.last.fm/. There are four classes, representing four quadrants of a valence-arousal plane: happy, sad, angry, and relaxed. The labels are unreliable and the classification task very hard, with previous research showing 54% classification accuracy as the top result. Despite that, we believe it represents a realistic scenario of musical data acquisition and the problems one may face when attempting to infer emotional content from unregulated tagging by a large community. Songs in the dataset are 30-second long excerpts.

Emotify game dataset [28], similarly to Lastfm, is based on crowd-sourced annotations, although the gathering process was much more controlled. Nine emotional labels represent nine middle emotions of GEMS, and the predicted values represent the percentage of users agreeing that particular emotion is induced by a particular music piece. It is important to note the explicit distinction between induced emotion versus perceived emotion. The dataset focuses on the former, and as a result, disagreement between annotators is very common. This disagreement is in part a result of variables that cannot be accounted for by music alone, including individual mood during annotation and individual connotations regarding specific words used to describe emotions. Because of that, predictions one can obtain through music audio analysis are poor on average: regarding Pearson's R coefficient, the correlation between predicted and actual values achieved in the first paper on this dataset was 0.47, averaged between all emotions (i.e., less than 25% of variance explained). However, there is a significant variance in figures of merit between specific emotions. For example, the emotion of amazement is almost unpredictable (below 0.3 correlation), while for joyful activation, Pearson's R above 0.7 is possible to achieve. Excerpts in the dataset are of varying lengths, 30- to 60-second long.

MediaEval 2013 data, also known as "1000 songs" dataset [29], is a set created for machine learning benchmarking. It consists of 744 (after duplicate removal) unique song excerpts. The dataset was annotated by crowd workers on Amazon Mechanical Turk platform, separately in valence and arousal dimensions, with each song receiving ten annotations. The publicly available data contains both mean and standard deviation of these evaluations. Songs in the dataset are 45-second excerpts.

GTZAN [30] is a famous dataset concerning genre recognition, and one of the most cited of all music information retrieval datasets. While it has been criticized for faults in its construction [31], as our research does not concern genre recognition specifically, we find it acceptable to use GTZAN for the sake of comparison between presented methods. GTZAN contains 30-second excerpts and is annotated as a classification dataset with ten genre labels.

Ballroom data [32] was originally meant for tempo estimation tasks. However, as the dataset offers clear split between classes corresponding to different dancing styles, we use these labels as a basis for a classification problem. It is interesting, as the distinction between dance styles is significantly more dependent on tempo and rhythm than the usual MIR tasks. Eight dance styles represented in the dataset are chacha, rumba, samba, quickstep, tango, slow waltz, and Viennese waltz.

2.5. Dataset Enhancement. As training of neural network models is strongly dependent on the quantity of available data, research datasets currently available for MIR tasks may pose the problem of insufficient files. We approach this issue using dataset enhancement, artificially expanding the number of possible training samples.

We use the following dataset enhancement scheme: during training, in each epoch instead of full feeding sequences corresponding to all music files in the dataset to the network, we choose a short excerpt of each file. This excerpt contains frames from t to $t + 100$ for a t randomly drawn from a uniform distribution.

Regarding dataset size, this approach hugely increases the number of potentially different samples a neural network will see during training. Consider a dataset of 1000 files, which are represented by sequences of 1200 vectors, 40 elements in a vector. These numbers correspond to spectrograms of 30-second files with extraction parameters employed in this article. Dimensions of the dataset are therefore $1000 \times 1200 \times 40$. However, with the enhancement, every 1200-frame long sequence has $1200 - 100 = 1100$ possible shorter excerpts of length 100. We are therefore effectively sampling from 1,100,000 possible excerpts that are sequences of 100 vectors, that is $1{,}100{,}000 \times 100 \times 40$.

It should be noted that the samples largely overlap and the network is likely to finish training before seeing every possible one. Additionally, this approach equates to learning on 2.5-second-long excerpts, thus ignoring any dependencies between frames separated by a time interval longer than 2.5 seconds within a music file. Nevertheless, the enhancement scheme allows us to test feature learning methods in a very efficient manner. We have previously shown [33] that this approach improves both convergence speed and the final result of the learning process for multiple music classification tasks when compared to training neural networks (both GRU and LSTM) on complete music files.

2.6. Common Conditions of the Experiments. For each of the performed experiments, we kept a similar core structure of the neural network and parameters for the said network. The network consists of two GRU recurrent layers, respectively, 100 and 50 units, and an additional nonrecurrent output layer in case of the baseline approach. The network is trained with Adadelta adaptive gradient descent method

[34]. Implementation of neural network logic and gradient operations uses the Theano library [35].

For SVM, we use an existing implementation from the scikit-learn python library. The hyperparameters of SVM were fit on the smallest dataset, Emotify, and reused in other experiments. Parameter C was set to 1, and the RBF kernel was employed with $\gamma = 0.5$. We ensured that on other datasets, a change in SVM parameters does not alter the results drastically, but we did not attempt further tuning the parameters for each dataset, as the resulting boost in performance was small, at the expense of creating unrealistic experimental conditions.

As an input to the feature extracting NN, we use a mel-frequency spectrogram with logarithmic power scale. The inputs are normalized to zero mean and standard deviation equal to 1 over each frequency bin, for each dataset independently. Frames of spectrogram are 50 ms long with 25 ms overlap, and we use 40 mel-frequency bins (the parameters were derived from defaults in MIRToolbox [36], a popular MATLAB toolbox for MIR feature extraction).

The input sequence to a recurrent network consists of 80-element vectors. These vectors contain the values of 40 spectrogram bins and the approximate of the first derivative (change from previous value) for each bin.

3. Results and Discussion

In the experiments, we seek to establish whether the proposed approach of learning a feature extracting neural network and supplying learned features to another classifier or regressor can improve results. As the main proposition of this paper is to adjust the learning goal (i.e., loss function) for a feature learning NN to one based on a notion of similarity as well as use a specific classifier on the learned features, we need two baselines for comparison. First one is a result of a full neural network-based approach, where the GRU network directly predicts classes or regressed variables. The second one is a result of an SVM deployed on bottleneck features (representation in a penultimate layer) from the aforementioned neural network. Altogether, we will compare five approaches:

(i) Baseline neural network approach (*NN*): GRU neural network approach in which the network is trained to classify or predict continuous values with standard sum square error loss.

(ii) SVM with baseline feature learning approach (*FL*): features are taken from the penultimate layer of the GRU neural network trained to classify or predict, then an SVM is trained on them.

(iii) SVM with NN learning RBF similarity (*RBF-SL*): feature extracting GRU neural network is learned with similarity-based loss using RBF kernel ($\gamma = 0.5$) for similarity, then SVM is trained on output features of the network.

(iv) SVM with NN learning cosine similarity (*Cos-SL*): feature extracting GRU neural network is learned with similarity-based loss using cosine similarity, then SVM is trained on output features of the network.

(v) SVM with NN learning polynomial similarity (*Poly-SL*): feature extracting GRU neural network is learned with similarity-based loss using polynomial kernel ($p = 2$, $b = 0$) for similarity, then SVM is trained on output features of the network.

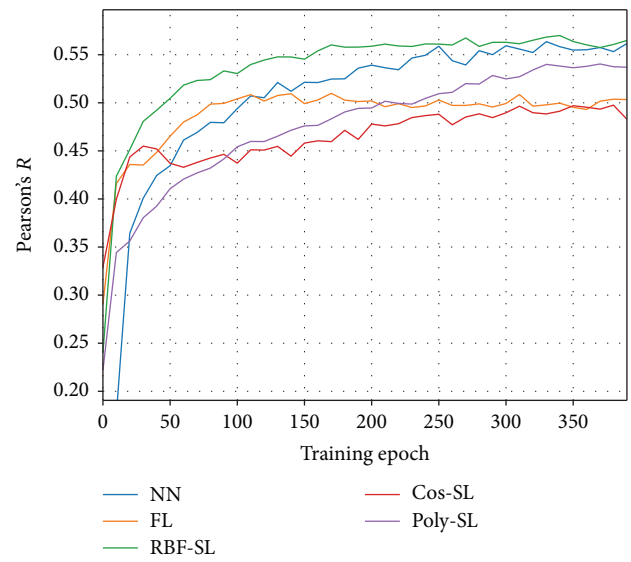

Figure 1: Emotify dataset, average prediction quality over all 9 GEMS emotions.

To demonstrate how the performance changes over the training process, we save the output of a feature extracting NN at every tenth epoch of its training. SVM is fully trained from zero at every one of these points to obtain a task-dependent measure of performance (accuracy for classification, Pearson's R for regression). We chose this way of presenting the results since, for the given dataset size, the time it takes to train SVM on learned features is a fraction of the time required to fully train a recurrent NN.

All presented results were obtained in 10-fold cross-validation experiments, in which the training-test split was applied to the learning of both the feature extracting NN and the SVM deployed on learned features afterwards.

3.1. Results on Emotion Regression Data. Results on emotion regression datasets are shown in Figures 1–3. On the plots of performance over the duration of training, we can see that the proposed approach with RBF kernel as a similarity measure achieves the best results and fastest convergence. This is consistent with our expectations, as RBF kernel was also used in the SVM model of regression. Compared to an SVM deployed on bottleneck features from a standard neural network, the loss function adjusted to learning similarity leads to improvements on all datasets. Compared to a purely NN approach, we can see either improvement or comparable performance. While cosine similarity measure consistently appears to perform the worst on the regression problems, it is hard to draw a comparison between polynomial kernel for similarity and SVM deployed on bottleneck features from

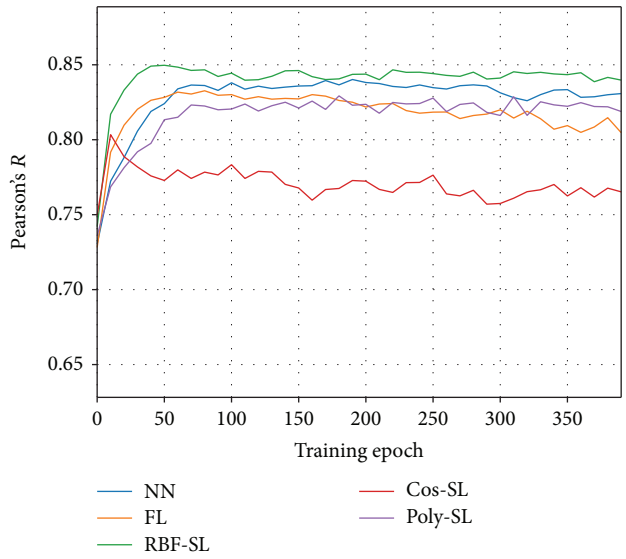

Figure 2: Performance on MediaEval2013 dataset, arousal prediction.

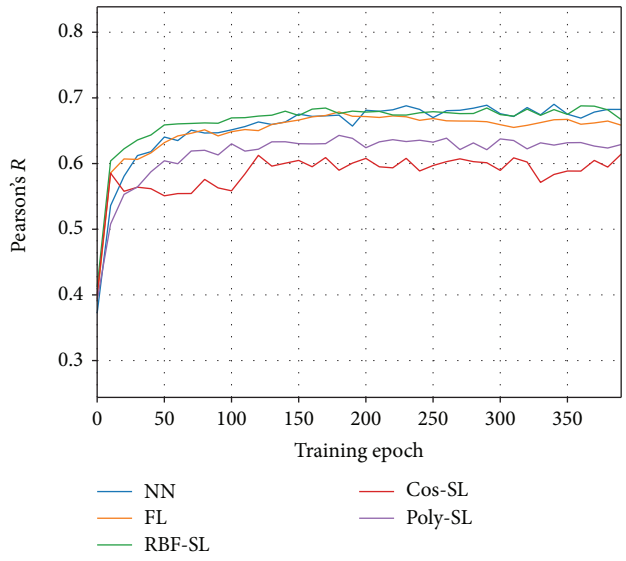

Figure 3: Performance on MediaEval2013 dataset, valence prediction.

regular NN. We can note that on the Emotify dataset, where using polynomial similarity yields improvement, the learning process is much slower than in other cases. Cosine similarity performs very badly on the MediaEval 2013 data, which can be explained by ignoring the norm of compared vectors. In V-A space, an emotional content labeled as 0.5 valence and 0.5 arousal will be vastly different than one labeled as 1 valence and 1 arousal.

In Table 2, we present the detailed results for recognition of specific emotions on the Emotify game dataset. The approach of learning RBF similarity within the feature extracting networks performs best for 7 out of 9 emotions. While results indicate a low quality of predictions, it can be noted that the proposed approach improves the worst results.

Table 2: Pearson's R on the Emotify dataset. Asterisk denotes SVM use.

Emotion	NN	FL*	RBF-SL*	Cos-SL*	Poly-SL*
Amazement	0.38	0.25	**0.40**	0.28	0.25
Solemnity	**0.56**	0.48	0.51	0.47	0.46
Tenderness	0.59	0.58	**0.62**	0.59	0.59
Nostalgia	0.58	0.56	**0.61**	0.53	0.56
Calmness	0.59	0.56	**0.60**	0.53	0.58
Power	0.54	0.49	**0.56**	0.52	0.52
Joyful act	0.69	0.67	**0.72**	0.65	0.68
Tension	**0.58**	0.56	0.57	0.51	0.56
Sadness	0.43	0.29	**0.48**	0.37	0.36

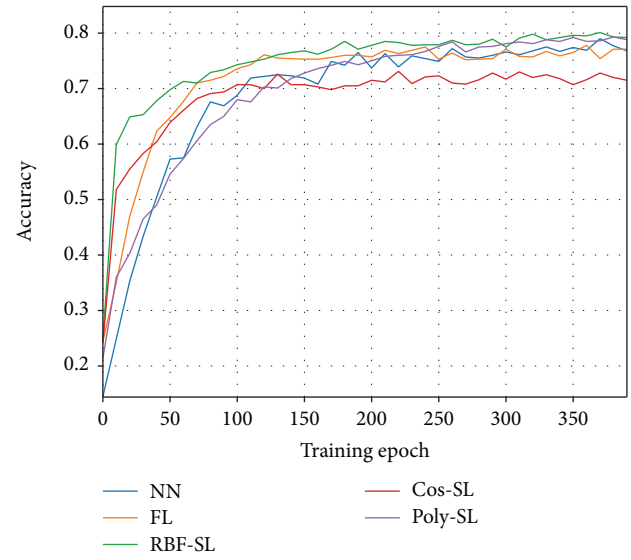

Figure 4: Classification performance on GTZAN dataset.

Notably, prediction of amazement reaches $R = 0.4$. This equates to the coefficient of determination $R^2 = 0.16$, that is, 16% of the variance in ground truth variable explained by the model. As previous results on the dataset indicated near-complete unpredictability of amazement category [11], the fact this one is above what is typically considered correlation by chance can be seen as relevant. For less subjective emotions, where making a prediction is more feasible, improvements can also be seen. In particular, the proposed approach is the only one where the model explains more than 50% of the variance for joyful activation ($R = 0.72$, $R^2 = 0.51$), but only if the similarity notion is chosen properly.

3.2. Results on Classification Tasks. Results on classification datasets GTZAN, ballroom, and Lastfm are shown in Figures 4–6. On the first two, we can see the proposed approach achieves the best performance with RBF kernel as a similarity measure.

In the proposed approach and SVM deployed on bottleneck features, the final result on the LastFM dataset is similar. However, the detailed look at the training process shows this is a result of overfitting on the part of the feature extracting

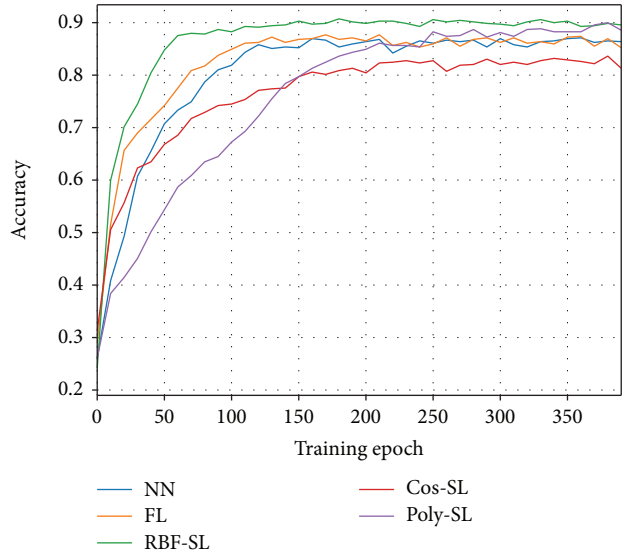

Figure 5: Classification performance on ballroom dataset.

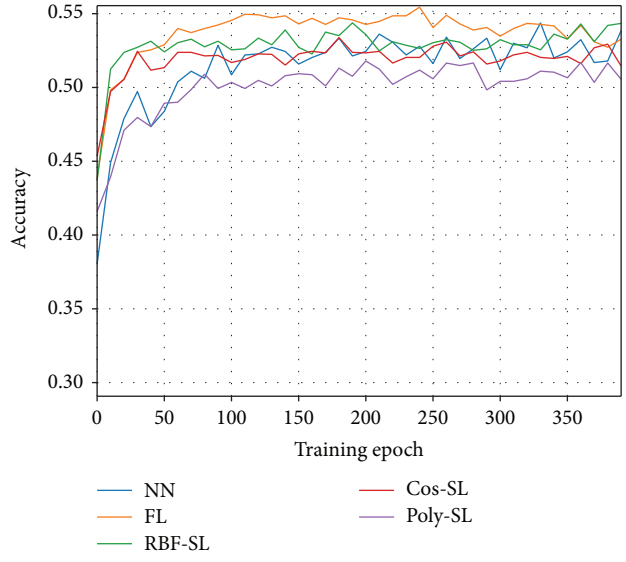

Figure 6: Classification performance on LastFM dataset.

NN, as in epochs 50–350 it achieves superior performance and only later drops to lower levels. It should be noted that LastFM data has the poorest quality of annotations among the examined datasets, which causes all of the tested approaches to be harder to evaluate.

Interestingly, polynomial kernel as a similarity measure appears to work significantly better on two of the tested classification datasets, and although it converges slowly, eventually it achieves the same results as RBF kernel on both GTZAN and ballroom data.

3.3. Statistical Significance of the Results. In cross-validation experiments, we observed that the proposed approach with appropriately selected similarity function either improves the baseline results or performs as well as the baselines on four of the examined datasets, while the results on LastFM data are inconclusive. To further confirm our conclusions rergarding the four datasets, we performed additional tests of best performing similarity function (RBF-SL) in comparison with regular NN and FL baseline approaches. These were repeated experiments on purely random 9 : 1 training-test split, without cross-validation, intended to gather a bigger sample size for testing the statistical significance of results. We repeated the random split experiment 100 times and tested the obtained results for statistical significance using Welch's t-test for unpaired samples with unequal variance. At the standard threshold ($p < 0.05$), we confirmed improvements after 400 epochs of training in comparison with NN baseline on datasets GTZAN and ballroom. In comparison with FL baseline, the improvements were confirmed on Emotify, as well as MediaEval2013. The RBF-SL approach did not perform worse than either of the two baselines in a statistically significant way on any of the datasets.

3.4. Conclusions and Future Work. From the presented results, we can conclude that the proposed approach of adjusting a loss function within the feature learning neural network to a similarity-based one can indeed improve the performance of an SVM later deployed on learned features. On all datasets, the proposed adjustment either outperforms purely NN-based approach or performs at least as well, when the learned notion of similarity is RBF kernel. This corresponds to the kernel used in the classifying SVM. When the learned notion of similarity is different, the performance can be vastly worse (cosine similarity), or comparable on some datasets, but worse on others (polynomial kernel). The modified loss function also shortens the learning of neural network feature extractor which, due to the complex nature of recurrent networks, is the most performance-demanding part of the learning process.

Our results are promising for the perspectives of future use for traditional machine learning approaches on musical data. While recent trends in machine learning focus on replacing older techniques with deep learning, in our experiments, best results are obtained when combining deep networks with a standard SVM approach. However, to achieve these results, the network has to be trained in a way that is adjusted to the specific classifier. A perspective for future research opens for creating similar adjustments targeting other standard machine learning approaches. One could also extend the possible future research to other types of data, where using deep learning on low-level representations is preferable to the extraction of features, for example, images and videos.

Data Availability

This study is based on previously reported data [27–30, 32]. As of writing the article (April 2018), music files and annotations for all of the examined datasets are available online at the following URLs: (i) Lastfm https://code.soundsoftware.ac.uk/projects/emotion-recognition, (ii) Emotify http://www.projects.science.uu.nl/memotion/emotifydata, (iii) MediaEval2013 http://cvml.unige.ch/databases/emoMusic, (iv) Ballroom http://mtg.upf.edu/ismir2004/contest/tempoContest/

node5.html, and (v) GTZAN http://marsyasweb.appspot.com/download/data_sets.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the statutory funds of the Department of Computational Intelligence, Faculty of Computer Science and Management, Wrocław University of Science and Technology.

References

[1] M. Defferrard, S. P. Mohanty, S. F. Carroll, and M. Salathe, "Learning to recognize musical genre from audio," 2018, https://arxiv.org/abs/1803.05337v1.

[2] C. C. Pratt, "Music as the language of emotion," *Bulletin of the American Musicological Society*, vol. 11, pp. 67-68, 1948.

[3] K. R. Scherer and M. Zentner, "Emotional effects of music: production rules," in *Music and Emotion: Theory and Research*, pp. 361–392, Oxford University Press, 2001.

[4] P. Ekman, "An argument for basic emotions," *Cognition and Emotion*, vol. 6, no. 3-4, pp. 169–200, 1992.

[5] Y. E. Kim, E. M. Schmidt, R. Migneco et al., "Music emotion recognition: a state of the art review," in *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, pp. 255–266, Utrecht, Netherlands, 2010.

[6] J. Skowronek, M. McKinney, and S. van de Par, "A demonstrator for automatic music mood estimation," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR 2007)*, pp. 345-346, Vienna, Austria, 2007.

[7] C. Laurier, O. Lartillot, T. Eerola, and P. Toiviainen, "Exploring relationships between audio features and emotion in music," in *Proceedings of the 7th Triennial Conference of European Society for the Cognitive Sciences of Music (ESCOM 2009)*, pp. 260–264, Jyväskylä, Finland, 2009.

[8] 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.

[9] 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.

[10] A. Nasridinov and Y. H. Park, "A study on music genre recognition and classification techniques," *International Journal of Multimedia and Ubiquitous Engineering*, vol. 9, no. 4, pp. 31–42, 2014.

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

[12] E. J. Humphrey, J. P. Bello, and Y. LeCun, "Feature learning and deep architectures: new directions for music informatics," *Journal of Intelligent Information Systems*, vol. 41, no. 3, pp. 461–481, 2013.

[13] U. Schimmack and R. Rainer, "Experiencing activation: energetic arousal and tense arousal are not mixtures of valence and activation," *Emotion*, vol. 2, no. 4, pp. 412–417, 2002.

[14] M. Henaff, K. Jarrett, K. Kavukcuoglu, and Y. LeCun, "Unsupervised learning of sparse features for scalable audio classification," in *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, pp. 681–686, Miami, FL, USA, 2011.

[15] Y. Vaizman, B. McFee, and G. Lanckriet, "Codebook-based audio feature representation for music information retrieval," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 10, pp. 1483–1493, 2014.

[16] J. Jakubik and H. Kwaśnicka, "Sparse coding methods for music induced emotion recognition," in *Proceedings of the 2016 Federated Conference on Computer Science and Information Systems*, pp. 53–60, Gdańsk, Poland, 2016.

[17] Y. Choi, G. Fazekas, M. Sandler, and K. Cho, "Convolutional recurrent neural networks for music classification," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2392–2396, New Orleans, LA, USA, 2017.

[18] J. Lee and J. Nam, "Multi-level and multi-scale feature aggregation using pretrained convolutional neural networks for music auto-tagging," *IEEE Signal Processing Letters*, vol. 24, no. 8, pp. 1208–1212, 2017.

[19] J. Jakubik and H. Kwaśnicka, "Music emotion analysis using semantic embedding recurrent neural networks," in *2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, pp. 271–276, Gdynia, Poland, 2017, IEEE.

[20] Y. Tang, "Deep learning using linear support vector machines," in *International Conference on Machine Learning 2013: Challenges in Representation Learning Workshop*, Atlanta, GA, USA, 2013.

[21] K. Choi, G. Fazekas, M. B. Sandler, and K. Cho, "Transfer learning for music classification and regression tasks," in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017)*, pp. 141–149, Suzhou, China, 2017.

[22] C. Goller and A. Kuchler, "Learning task-dependent distributed representations by backpropagation through structure," in *Proceedings of International Conference on Neural Networks (ICNN'96)*, pp. 347–352, Washington, DC, USA, 1996.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations (ICLR 2015)*, San Diego, CA, USA, 2015.

[25] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, https://arxiv.org/abs/1412.3555.

[26] H. Wu, M. R. Min, and B. Bai, "Deep semantic embedding," in *Proceedings of Workshop on Semantic Matching in Information Retrieval co-located with the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval (SMIR@SIGIR 2014)*, pp. 46–52, Gold Coast, QLD, Australia, 2014.

[27] Y. Song, S. Dixon, and M. Pearce, "Evaluation of musical features for emotion classification," in *Proceedings of the 13th*

International Society for Music Information Retrieval Conference (ISMIR 2012), pp. 523–528, Porto, Portugal, 2012.

[28] A. Aljanaki, F. Wiering, and R. C. Veltkamp, "Studying emotion induced by music through a crowdsourcing game," *Information Processing & Management*, vol. 52, no. 1, pp. 115–128, 2016.

[29] M. Soleymani, M. N. Caro, E. M. Schmidt, C. Y. Sha, and Y. H. Yang, "1000 songs for emotional analysis of music," in *Proceedings of the 2nd ACM International Workshop on Crowdsourcing for Multimedia - CrowdMM '13*, Barcelona, Spain, 2012.

[30] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.

[31] B. L. Sturm, "The GTZAN dataset: its contents, its faults, their effects on evaluation, and its future use," 2013, https://arxiv.org/abs/1306.1461.

[32] K. Seyerlehner, G. Widmer, and D. Schnitzer, "From rhythm patterns to perceived tempo," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR 2007)*, pp. 519–524, Vienna, Austria, 2007.

[33] J. Jakubik, "Evaluation of gated recurrent neural networks in music classification tasks," in *Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture And Technology, ISAT 2017* of *Advances in Intelligent Systems and Computing*, pp. 27–37, Szklarska Poręba, Poland, 2018.

[34] M. D. Zeiler, "ADADELTA: an adaptive learning rate method," 2012, https://arxiv.org/abs/1212.5701.

[35] Theano Development Team, "Theano: a python framework for fast computation of mathematical expressions," 2016, https://arxiv.org/abs/1605.02688.

[36] O. Lartillot and P. Toiviainen, "A Matlab toolbox for musical feature extraction from audio," in *International Conference on Digital Audio Effects (DAFX 2018)*, pp. 237–244, Aveiro, Portugal, 2007.