Recognizing Song Mood and Theme Using Convolutional Recurrent Neural Networks

Maximilian Mayerl[†1], Michael Vötter[†1], Hsiao-Tzu Hung[2],
Bo-Yu Chen[3], Yi-Hsuan Yang[2,3] Eva Zangerle[1]
[1]Universität Innsbruck, Austria
[2]Taiwan AI Labs, Taiwan,
[3]Research Center for IT Innovation, Academia Sinica, Taiwan
maximilian.mayerl@uibk.ac.at,michael.voetter@uibk.ac.at,fbiannahung@gmail.com
bernie40916@gmail.com,affige@gmail.com,eva.zangerle@uibk.ac.at

ABSTRACT

In this year's MediaEval task, *Emotion and Theme Recognition in Music Using Jamendo*, the goal is to assign emotion and theme tags to songs. In this paper, we describe our–Team TaiInn (Innsbruck)– approach for this task. We use a neural network model consisting of both convolutional and recurrent layers and utilize spectral, high-level as well as rhythm features. Our approach achieves a ROC-AUC score of 0.723 on the provided test set.

1 INTRODUCTION

At this year's MediaEval workshop, the task *Emotion and Theme Recognition in Music Using Jamendo* deals with detecting mood and theme tags for songs that are described by audio descriptors. Those songs are drawn from Jamendo[1], and collected in a data set created by Bogdanov et al. [3]. The tags that have to be predicted cover a wide range of emotions and themes, and include tags like *sad*, *heavy*, *relaxing*, and *children*. More details can be found in [1].

We use a neural network model to approach this task and show that our model achieves a performance that is comparable to the provided baseline (ROC-AUC of 0.723 for our best run), but requires much fewer training epochs to reach this (16 vs 1,000). We also show that generating more training samples by drawing random windows from the provided mel-spectrograms improves results, and that incorporating high-level features into the model architecture reduces the number of needed epochs in training while giving almost the same results. Our code is available on GitHub[2].

2 APPROACH

Our approach to the task is an extension to the CRNN (convolutional recurrent neural network) model proposed by Choi et al. [4]. Hence, we utilize a neural network consisting of a combination of convolutional, recurrent as well as dense blocks. The network is structured in a way that allows to efficiently combine different types of features, similar to the approach used by Zangerle et al. [6] for combining low- and high-level features in a neural network model for the task of hit song prediction.

In the remainder of this section, we will explain our approach to the task in detail. Section 2.1 describes how we use the provided data to train our models. Section 2.2 explains the structure of our model. Finally, Section 2.3 details the runs we submitted for the task.

2.1 Data

For training our models, we used both the precomputed mel-spectrograms and audio features computed with Essentia [2] that were provided by the task organizers. From those, we utilized the mel-spectrograms as well as high-level features (like genre and danceability), and rhythm features (beats per minute). As training data, the task organizers provided a set of 9,949 songs. We used two different training sets. The *Base* training set is the set provided by the task organizers. For the mel-spectrograms, we use a window of width 1,366 around the temporal center of the song. For the *RandomSampling* training set, on the other hand, we generated more training samples by taking five random windows of width 1,366 from each song. This training set therefore provides us with 49,745 training samples.

2.2 Model

Our approach uses a neural network model consisting of convolutional, recurrent, and dense layers. The mel-spectrograms are fed into a series of convolutional layers, followed by two recurrent layers using GRU (gated recurrent unit) cells. High-level and rhythm features bypass this part of the network and directly feed into the dense part that comes after the recurrent layers. A schematic overview of the base network structure is given in Figure 1. Some of our submissions feature modifications to this structure. Those modifications are described in more detail in Section 2.3.

Every convolutional block consists of a 2D convolutional layer followed by batch normalization, `ELU` activation and a max pooling layer. After every block, a dropout of 10% is applied. The convolutional layers have kernel sizes of `(3, 3)`, whereas the pooling layers have pool sizes of `(2, 2)`, `(3, 3)`, `(4, 4)`, and `(4, 4)`. The first convolutional layer uses 64 filters, whereas the latter three layers use 128 filters. This is followed by two recurrent layers using GRU cells. Both of these layers use `tanh` activation. Following that, the output of the second GRU layer is concatenated with the high-level and rhythm features and fed into two dense layers. The first dense layer has 128 units and uses `tanh` activation, whereas the second and final dense layer has 56 units—the number of possible

[†]authors contributed equally to this work
[1]https://www.jamendo.com
[2]https://github.com/dbis-uibk/MediaEval2019

MediaEval'19, 27-29 October 2019, Sophia Antipolis, France

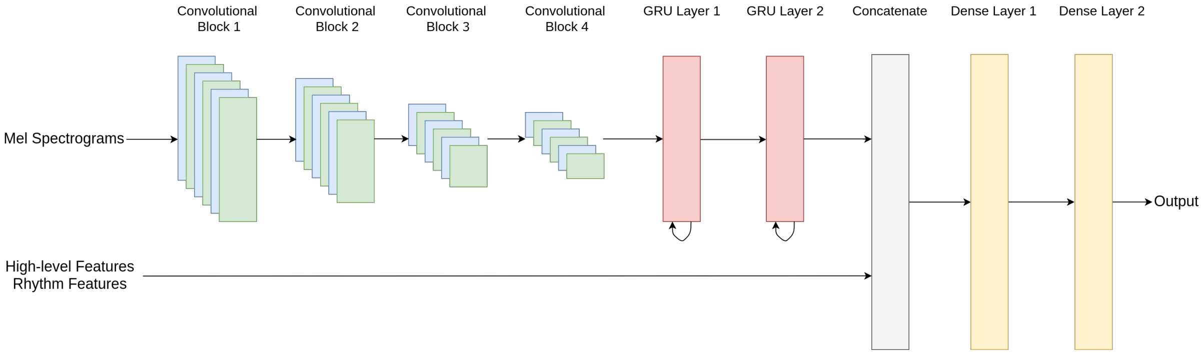

Figure 1: Schematic of our network structure.

labels—and uses `sigmoid` activation. A dropout of 30% is applied after the first of the dense layers.

The model is trained using the Adam optimizer [5]. As loss function, we used binary cross-entropy. The output of this model is a vector of probabilities, where every entry in the vector holds the probability for one of the labels. To then decide which labels to assign to a given song, we determine the best probability thresholds for every label individually, using the provided validation set. For this, we use a variation of the elbow method on the ROC curve: We compute the per label ROC curve on the given validation set. Based on this ROC curve we compute the straight line between the left most point and the right most point. This line is used as a reference to find the point on the ROC curve that has the largest orthogonal distance to that line. We then use the corresponding threshold as our decision boundary for that label.

2.3 Submissions

Using the data and model described above, we build multiple submission runs:

- *Run #1*: The base model, but without the high-level and rhythm features, trained on the *Base* training set.
- *Run #2*: The base model, but without the high-level and rhythm features, trained on the *RandomSampling* training set.
- *Run #3*: The base model, but without the high-level and rhythm features, and with an attention mechanism after the second recurrent layer, trained on the *RandomSampling* training set.
- *Run #4*: The base model, as described in Section 2.2, trained on the *RandomSampling* training set.
- *Run #5*: The base model, with an attention mechanism after the second recurrent layer, trained on the *RandomSampling* training set.

3 RESULTS AND ANALYSIS

The results for the submissions described in Section 2.3, as well as how many epochs were used for training the respective model, are summarized in Table 1. As can be seen, all of our runs outperform the popularity baseline, but do not manage to beat the VGG-ish baseline. The best performing runs are run #2 and #3, which both perform very close to the VGG-ish baseline in terms of ROC-AUC and PR-AUC, but significantly worse in terms of F_1 score. On the other hand, our runs require much fewer training epochs to achieve those results. The fact that ROC-AUC and PR-AUC are similar but F_1 is worse suggests that our choice of probability thresholds for label assignments is suboptimal.

Table 1: Evaluation results for our runs and for the provided baselines. Submitted runs are shown in bold.

Run	Epochs	ROC-AUC	PR-AUC	F_1 (micro)	F_1 (macro)
Popularity	-	*0.500*	*0.032*	*0.057*	*0.003*
VGG-ish	*1,000*	*0.726*	*0.108*	*0.177*	*0.166*
#1	8	0.549	0.043	0.071	0.071
#1	**16**	**0.700**	**0.080**	**0.103**	**0.106**
#2	**16**	**0.719**	**0.110**	**0.104**	**0.114**
#3	**16**	**0.723**	**0.110**	**0.108**	**0.111**
#4	8	0.650	0.069	0.083	0.085
#4	**16**	**0.685**	**0.090**	**0.095**	**0.101**
#5	**16**	**0.684**	**0.089**	**0.099**	**0.102**

Comparing the results for our runs which use both spectrograms as well as high-level features as input (#4, #5) with the runs that use only mel-spectrograms (#1, #2, and #3), we observe that the models without high-level features seem to perform better overall. Despite this, the models using high-level features learn faster, with the difference between 8 and 16 epochs of training being much smaller for run #4 than it is for run #1.

We can also see that random sampling for increasing the number of training samples helps to improve results. The PR-AUC increases from 8% in run #1 to 11% in run #2, which only differ in that run #2 uses random sampling while run #1 does not.

4 SUMMARY AND OUTLOOK

In this paper, we described our approach to the *Emotion and Theme Recognition in Music Using Jamendo* task at MediaEval 2019. Our best approach achieved a ROC-AUC score of 0.723, which is slightly worse than the best baseline provided by the task organizers (0.726), but needs much fewer training epochs than the baseline. Potential future work on this approach includes trying different ways to determine probability thresholds for label assignments and applying this approach to other data sets for the same task.

REFERENCES

[1] Dmitry Bogdanov, Alastair Porter, Philip Tovstogan, and Minz Won. 2019. MediaEval 2019: Emotion and Theme Recognition in Music Using Jamendo. In *CEURS Working Notes Proceedings of the MediaEval 2019 Workshop*. CEUR-WS.org.

[2] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez Gutiérrez, Sankalp Gulati, Perfecto Herrera Boyer, Oscar Mayor, Gerard Roma Trepat, Justin Salamon, José Ricardo Zapata González, and Xavier Serra. 2013. Essentia: An audio analysis library for music information retrieval. In *Britto A, Gouyon F, Dixon S, editors. 14th Conference of the International Society for Music Information Retrieval (ISMIR); 2013 Nov 4-8; Curitiba, Brazil.[place unknown]: ISMIR; 2013. p. 493-8.* International Society for Music Information Retrieval (ISMIR).

[3] Dmitry Bogdanov, Minz Sanghee Won, Philip Tovstogan, Alastair Porter, and Xavier Serra. 2019. The MTG-Jamendo Dataset for Automatic Music Tagging. (2019).

[4] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. 2017. Convolutional recurrent neural networks for music classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2392–2396.

[5] Diederik Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations* (12 2014).

[6] Eva Zangerle, Michael Vötter, Ramona Huber, and Yi-Hsuan Yang. 2019. Hit Song Prediction: Leveraging Low- and High-Level Audio Features. In *Proceedings of the 20th International Society for Music Information Retrieval Conference 2019 (ISMIR 2019)*.