# MediaEval 2019 Emotion and Theme Recognition task: A VQ-VAE Based Approach

Hsiao-Tzu Hung[†1], Yu-Hua Chen[†1],Maximilian Mayerl[3],
Michael Vötter[3] Eva Zangerle[3], Yi-Hsuan Yang[1,2]
[1]Taiwan AI Labs, [2]Research Center for IT Innovation, Academia Sinica, Taiwan, [3]Universität Innsbruck, Austria
fbiannahung@gmail.com,r08946011@ntu.edu.tw,maximilian.mayerl@uibk.ac.at
michael.voetter@uibk.ac.at,eva.zangerle@uibk.ac.at,affige@gmail.com

## ABSTRACT

In this paper, we, Taiinn (Taiwan) team, use pre-trained VQ-VAE as a feature extractor and compare two types of classifier for audio-based emotion and theme recognition. The VQ-VAE is pre-trained on the Million Song Dataset (MSD). We found better performance in ROC-AUC by fixing the pre-trained parameters of VQ-VAE while training the classifier. In addition, an embedding with bigger shape works better than the one-dimensional counterpart. The code and submitted models can be found at: https://github.com/annahung31/moodtheme-tagging.

## 1 INTRODUCTION

This paper describes our submission to the MediaEval 2019 Emotion and Theme recognition task [2]. The goal is to automatically assign audio clips with emotion and theme tags using a data collection from Jamendo, a platform of copyright free music. The task can be considered as a multi-label, music auto-tagging problem [6].

Lately, vector-quantized variational auto-encoder (VQ-VAE) [8] has been shown effective for images and audio generation. It learns a quantized representation of its input in an unsupervised way. This motivates us to study the use of VQ-VAE for classification problems such as the one involved in the MediaEval 2019 Emotion and Theme task. While our work remains preliminary, it seems no previous work has used VQ-VAE for auto-tagging problems.

## 2 APPROACH

### 2.1 Third-party dataset

Besides the Jamendo dataset prepared by the task organizers, we also use the million song dataset (MSD) [1] and the MagnaTagATune (MTAT) dataset [4] in our work. The number of samples of the two datasets can be found in Table 1. We use MSD only for pre-training the VQ-VAE model, so we only split the datset into training and validation sets. As for MTAT, we use it as the second test set (in addition to Jamendo) for testing VQ-VAE, and hence we split it into training, validation, and test sets. We only consider the top-50 tags (mostly genre and instrument tags [3]) for MTAT.

[†]The two authors contributed equally to this work


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

**Table 1: Number of audio samples of third-party datasets in the train, validation and test splits we made**

| | Train | Validation | Test |
|---|---|---|---|
| MSD [1] | 557,315 | 37,008 | 0 |
| MTAT [4] | 16,776 | 1,339 | 2,651 |

### 2.2 Input feature

We use librosa [5] to extract 128-dimensional log-mel spectrums from the audio files. The sampling rate is set to be 22,050 Hz, and only first 1,024 frames are took for every clips, leading to a fixed-size matrix of $128 \times 1024$ per clip.

### 2.3 Neural networks

*2.3.1 VQ-VAE as feature extractor.* We use VQ-VAE as an feature extractor to get a discrete embedding from mel-spectrograms. The VQ-VAE basically contains an encoder and a decoder. The encoder contains 5 convolutional layers, followed by two residual 3×3 blocks all having 256 feature maps. The kernel size and the stride of the first 4 layers is (4,3), (2,1), and those of the fifth layer are (5,4), (1,2). The padding of every layer are (1,2), (1,4) ,(1,8), (1,16), (0,1). The dilation are the same as padding. As a result, the encoder will generate an embedding with shape of $256 \times 4 \times 512$. The decoder consists two residual $3 \times 3$ blocks, followed by 5 transposed convolutional layers. The kernel size, stride and padding for the first later is (4,4), (1,2), (0,1), and are (4,3), (2,1), (0.1) for the second layer. For the remaining three layers, the kernel size, stride and padding are (4,3), (2,1), (1,1). In the end of the decoder, an activation function of tanh is used. We call the this Type-1 VQ-VAE.

To observe how the dimension of the embedding affects the performance of tagging, we implement an alternative that uses (8,4) kernel for the fifth layer of the encoder, making the shape of the embedding $256 \times 1 \times 512$. We may view it as a sequence of 256-dimensional feature vectors. We call this one Type-2 VQ-VAE.

*2.3.2 Classifiers.* We use two kinds of classifier for training. The first one is a GRU-classifier, with 2 bi-directional gated recurrent units (GRUs). After the first GRU, layer normalization is applied. The output hidden states of the second GRU will then go through a fully-connected layer and sigmoid activation layer to get prediction. The second one is a CNN (convolutional neural network)-classifier. The model structure of the CNN classifier is basically the same as that proposed in [7], with the size of channels halved.

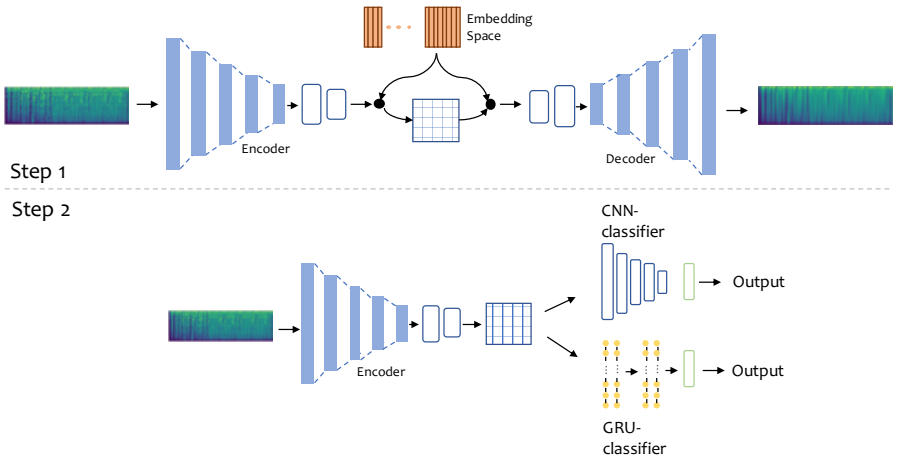


**Figure 1: Schematic architecture of the proposed neural network and training procedure.**

## 2.4 Training

The training procedure, as depicted in Figure 1, is composed of two steps. In step-1, we pre-train VQ-VAE on MSD by minimizing the reconstruction error. In step-2, we cascade the encoder of VQ-VAE trained in step-1 along with a classifier (a GRU or a CNN based one), and train the network by binary cross entropy loss for genre, mood or theme recognition (depending on the dataset). During the training process, we set the batch size to 12 and learning rate to 2e-4. The Adam optimizer is used to train the models. The networks are trained for a maximum of 100 epochs with early stopping.

## 2.5 Methods

We submit the following five runs:

- **Run-1**: type-1 VQ-VAE + GRU; updating both VQ-VAE and GRU during step-2 training.
- **Run-2**: type-1 VQ-VAE + GRU; fixing VQ-VAE and updating only the GRU during step-2 training.
- **Run-3**: type-1 VQ-VAE + CNN; updating both VQ-VAE and CNN during step-2 training.
- **Run-4**: type-1 VQ-VAE + CNN; fixing VQ-VAE and updating only the CNN during step-2 training.
- **Run-5**: type-2 VQ-VAE + GRU; updating both VQ-VAE and GRU during step-2 training.

# 3 RESULTS AND ANALYSIS

## 3.1 Auto-tagging on MTAT

To verify the effectiveness of the VQ-VAE based classification method, we firstly evaluate the run-1 method on MTAT for auto-tagging. Specifically, in step-2 training, we update the type-1 VQ-VAE (pre-trained on MSD) along with the GRU classifier on MTAT and observe the performance of tagging. It turns out that the model attains ROC-AUC 0.90 when predicting top-50 tags, which is close to the performance of state-of-the-art models [6].

**Table 2: Testing (first seven rows) and validation (last five) scores on the MediaEval'19 Jamendo dataset.**

| | ROC-AUC | PR-AUC | F1(macro) | F1(micro) |
|---|---|---|---|---|
| Popularity | 0.5000 | 0.0320 | 0.0570 | 0.0030 |
| VGG-ish | 0.7258 | 0.1077 | 0.1657 | 0.1771 |
| Run-1 | 0.7103 | 0.0984 | 0.1183 | 0.1439 |
| Run-2 | 0.7141 | 0.1037 | 0.0901 | 0.1184 |
| Run-3 | 0.7147 | 0.0994 | 0.1013 | 0.1233 |
| Run-4 | 0.7207 | 0.1077 | 0.1068 | 0.1522 |
| Run-5 | 0.6916 | 0.0860 | 0.0884 | 0.1209 |
| Run-1 | 0.6829 | 0.0717 | 0.0891 | 0.1161 |
| Run-2 | 0.6973 | 0.0782 | 0.0838 | 0.1201 |
| Run-3 | 0.6928 | 0.0746 | 0.0921 | 0.1227 |
| Run-4 | 0.6966 | 0.0770 | 0.0851 | 0.1142 |
| Run-5 | 0.6662 | 0.0608 | 0.0746 | 0.0899 |

## 3.2 Mood & theme classification on Jamendo

The result on the Jamendo dataset is shown in Table 2. We can see that, in terms of ROC-AUC, Run-2 outperforms Run-1, and Run-4 outperforms Run-3. This may indicate that it is better to fix the VQ-VAE when training the classifiers. We can also see that the CNN classifier seems to perform slightly better than the GRU classifier. And, it seems that the type-1 VQ-VAE works than the type-2 counterpart. The best ROC-AUC 0.7207 is obtained by Run-4. Yet, it is worse than VGG-ish, which represents a strong baseline.

# 4 SUMMARY AND OUTLOOK

In this paper, we have reported a preliminary attempt that uses pre-trained VQ-VAE model for music auto-tagging problems. From the evaluation result, it seems that either the approach is not that promising for discrminative tasks, or that we have not fully capitalized its potential. We would like to further develop this approach in the near future, for both discrminative and generative problems in music (e.g., to generate music in the audio domain).

## REFERENCES


[1] Thierry Bertin-Mahieux, Daniel P.W. Ellis, and and Paul Lamere Brian Whitman. 2011. The million song dataset. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*.

[2] Dmitry Bogdanov, Alastair Porter, Philip Tovstogan, and Minz Won. 2019. MediaEval 2019: Emotion and theme recognition in music using Jamendo. In *MediaEval 2019 Workshop*.

[3] Keunwoo Choi. 2017. List of automatic music tagging research articles that are evaluated against MagnaTagATune Dataset. https://github.com/keunwoochoi/magnatagatune-list. (2017). Online; accessed 29 September 2019.

[4] Edith Law, Kris West, Michael I. Mandel, Mert Bay, and J. Stephen Downie. 2009. Evaluation of algorithms using games: The case of music tagging. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*.

[5] Brian McFee, Colin Raffel, Dawen Liang, Daniel P. W . Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. 2015. librosa: Audio and music signal analysis in Python. In *Proc. Python in Science Conf.* 18–25. [Online] https://librosa.github.io/librosa/.

[6] Juhan Nam, Keunwoo Choi, Jongpil Lee, Szu-Yu Chou, and Yi-Hsuan Yang. 2019. Deep learning for audio-based music classification and tagging: Teaching computers to distinguish rock from Bach. *IEEE Signal Processing Magazine* 36, 1 (2019), 41–51.

[7] Jordi Pons, Oriol Nieto, Matthew Prockup, Erik M. Schmidt, Andreas F. Ehmann, and Xavier Serra. 2018. End-to-end learning for music audio tagging at scale. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*.

[8] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural discrete representation learning. In *Proc. Conference on Neural Information Processing Systems (NIPS)*.