Automatic music tagging with music-domain text descriptions

COMP 5221 • Natural Language Processing

Kevin Zhang • Zusheng Tan • Hanyu Duan
Prof. Dekai Wu
May 29, 2023

Abstract

The development of automatic music tagging systems has been hindered by data scarcity and reliance on a rigid set of predefined music attributes. To address this challenge, we propose CNN-LSTM-Autoencoder Music Tagging (CLSTMA-MT), a model that generates detailed captions encapsulating the essence of music recordings. The model adopts an autoencoder structure, integrating LSTM and CNN components trained on the MusicCaps dataset. We establish multiple baselines to evaluate the output quality, with the model achieving the highest performance, yielding a cross-entropy loss of 2.89. The versatility of our system enables its application across various music-related domains, such as audio-based music retrieval, music recommendation, playlist generation, and music analysis. The source code is publicly available at: https://github.com/allent4n/auto-music-tag.

1 Introduction

Music tagging is a significant task in Music Information Retrieval (MIR) that involves assigning descriptive labels or tags to music or audio recordings. It plays a crucial role in extracting valuable information from audio, including genre, mood, instrumentation, tempo, and other characteristics. This process typically entails extracting acoustic features representing the desired tags, followed by single or multi-label classification.

However, traditional methods face limitations in the classification phase, as they rely on predefined music classes to assign audio samples to specific categories based on the extracted features [4, 5, 6, 8]. These limitations are exacerbated by the scarcity of audio-text data, particularly in comparison to the abundance of image-text data, which significantly contributes to image generation quality [1]. Unlike images, describing audio with concise and unambiguous text is challenging, as capturing the salient characteristics of acoustic scenes or music in a few words is not straightforward. Furthermore, the temporal nature of audio introduces complexities in creating sequence-wide captions, which are weaker in providing comprehensive annotations compared to image captions [2].

To address the current challenges in music tagging, we propose a system called CNN-LSTM-Autoencoder Music Tagging (CLSTMA-MT). Our system leverages the MusicCaps dataset, a valuable resource consisting of audio-text pairs labeled by professional musicians. The CLSTMA-MT system follows an autoencoder structure, incorporating a combination of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks during the encoding phase to learn a latent representation of the audio data. The decoder utilizes an LSTM layer to reconstruct the latent representation into meaningful text captions. We evaluate the performance of our system against various baseline models to assess its effectiveness in generating informative and accurate captions for music audio.

2 Related work

Our project was inspired mainly by Google's MusicLM, a conditional Multimodal Generative Transformer (MGS) model that combines descriptive text and melodies to generate music sequences [2]. It can generate longer sequences by using a 15-second segment of the generated music as a prefix to generate additional 15-second music segments. The model also incorporates a "Story mode" feature, generating music based on changing text descriptions every 15 seconds, creating a dynamic musical narrative.

To ensure overall model stability and simplicity, we also drew inspiration from conventional automatic music tagging systems that employed popular and effective neural network architectures: CNNs, RNNs, GRUs, and LSTMs. These architectures exhibit similarities in their neural network structure, usage of nonlinear activation functions, learnable parameters, and feature extraction capabilities. While RNNs, GRUs, and LSTMs excel in capturing sequential dependencies and temporal patterns, CNNs detect local patterns and spatial relationships. These architectural variances allow deep learning models to learn hierarchical representations and effectively model complex relationships in data. The specific distinctions of each model will be elaborated on in the subsequent sections.

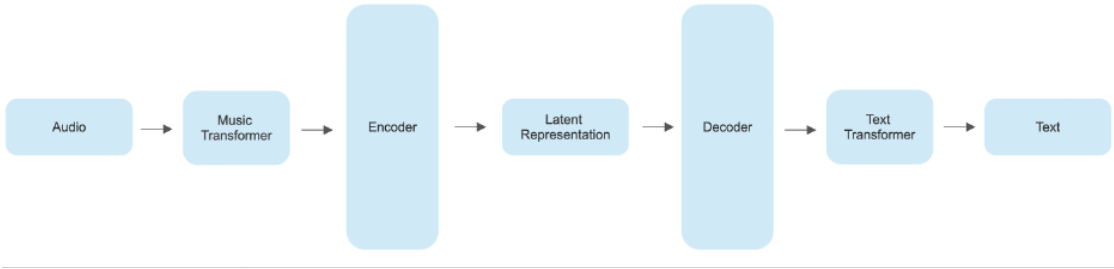
3 Proposed approach

3.1 Dataset

The MusicCaps dataset consists of 5,521 ten-second music audio samples tagged with an aspect list labeled by professional musicians. The aspect list is more subjective and descriptive while incorporating more audio and music-related terminologies, e.g., [‘low quality,’ ‘noisy,’ ‘mono,’ ‘harmonizing mixed vocals’]. The dataset was released from Google to supplement MusicLM to support future MIR research.

3.2 Model architecture

A mapping between audio-based and text-based vectors and text-based vectors must be established to enable the model to generate text or aspects when presented with music audio automatically. To accomplish this, we utilize an autoencoder architecture designed to learn a latent representation of the data by approximating the input to the output. An autoencoder comprises two main parts: an encoder and a decoder.



The encoder part of an autoencoder takes the input $\mathbf{x} \in \mathbb{R}^d$ and maps it to a latent (hidden) representation $\mathbf{x} \in \mathbb{R}^d$, where d and p are the dimensionalities of the input data and latent representation, respectively, which is often $p << d$. The mapping is achieved through a function f_θ parameterized by θ, such as a neural network

$$\mathbf{z} = f_\theta(\mathbf{x}) .$$

The decoder part of the autoencoder takes the latent representation $\mathbf{z}$ and maps it back to the reconstructed input $\mathbf{x}'$ in the original input space. This mapping is typically achieved through a function g_ϕ, also parameterized by ϕ:

$$\mathbf{x}' = g_\phi(\mathbf{z})\,.$$

The goal of an autoencoder is to learn the parameters θ and ϕ that minimizes the difference between $\mathbf{x}$ and $\mathbf{x}'$, measured by a loss function. The cross-entropy loss is used in this project, and the definition for a single example is:

$$L(\mathbf{x}, \mathbf{x}') = -\sum_i [x_i \log x_i' + (1 - x_i)\log(1 - x_i')]$$

where x_i and x_i' are the i-th components of $\mathbf{x}$ and $\mathbf{x}'$. For a batch of N examples, the cross-entropy loss is averaged over all examples in the batch:

$$L(\mathbf{x}, \mathbf{x}') = -\frac{1}{N}\sum_{n=1}^{N}\sum_i [x_{n,i} \log x_{n,i}' + (1 - x_{n,i})\log(1 - x_{n,i}')]\,.$$

3.3 Baselines

In music tagging, the autoencoder component encodes the music vector into a latent representation and subsequently decodes this representation into a text vector. To enhance the representation of both the input music audio and the output text, various models are incorporated in the encoder-decoder components to improve the quality and accuracy of the learned representations.

3.3.1 Recurrent neural network

Recurrent Neural Networks (RNNs) and their variants, such as Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTM) networks, are commonly used in music tagging tasks due to their capability to capture sequential dependencies and temporal patterns in music data [4, 5, 13]. These models are particularly well-suited for modeling music audio, which exhibits a temporal structure. Among the RNN variants, LSTM networks have gained recognition for modeling long-term dependencies by incorporating specialized memory cells that selectively forget or retain information over time. This characteristic enables LSTMs to overcome the issues of vanishing or exploding gradients often encountered with traditional

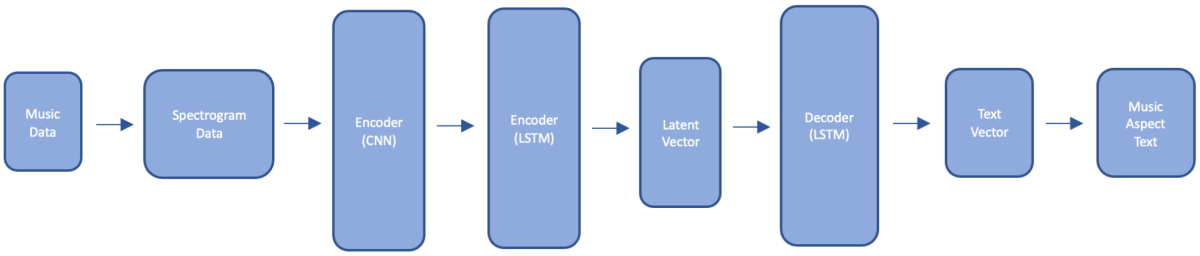
RNNs during training. GRUs, on the other hand, share similarities with LSTMs in their use of gating units to control the flow of information. However, GRUs provide a computationally simpler alternative by utilizing two gates instead of three and combining the memory cell and hidden state into a single hidden state, contributing to increased efficiency compared to LSTMs.

3.3.2 Convolutional neural network

Convolutional neural networks (CNNs) have emerged as a common and effective approach for music tagging [6, 8, 9, 11, 12, 14]. The inspiration behind CNNs originates from the biological vision systems, where local information is captured by multiple sensory cells and used to extract higher-level understanding [7]. This concept is translated into learning robust features that remain the same even if things change, move around, or get distorted, similar to how our eyes and brain understand what we see [8]. Leveraging the topographical representation of audio signals, which depict the acoustical frequencies over time, CNNs can extract pertinent features such as spectral patterns, harmonics, and timbre, contributing to accurate audio content classification [6].

3.3.3 Hybrid models

Autoencoders can be composed of different models to leverage the strengths of each model to achieve effective feature extraction and reconstruction in the autoencoder framework [10, 11, 13]. After experimenting with several combinations of different models, a hybrid CNN and LSTM achieved the best results.



A high-level mathematical definition of CLSTMA-MT is as follows:

1. *Convolution layer*: For an input tensor $\mathbf{X} \in \mathbb{R}^{\text{batch size} \times \text{input size} \times \text{sequence length}}$, the operation performed by the convolution layer is

$$\mathbf{Z}_{\text{conv}} = f(\mathbf{X} * \mathbf{W}_{\text{conv}} + \mathbf{b}_{\text{conv}})$$

where $\mathbf{W}_{\text{conv}} \in \mathbb{R}^{\text{input size} \times \text{hidden size} \times \text{kernel size}}$ are the convolutional weights, $\mathbf{b}_{\text{conv}} \in \mathbf{R}^{\text{hidden size}}$ are the biases, and f is the ReLU activation function.

2. *Encoder LSTM*: For the output of the convolutional layer $\mathbf{Z}_{\text{conv}}$, the operation performed by the LSTM is

$$(\mathbf{H}_{\text{encoder}}, \mathbf{C}_{\text{encoder}}) = \text{LSTM}_{\text{encoder}}(\mathbf{Z}_{\text{conv}}, (\mathbf{H}_{\text{encoder},0}, \mathbf{C}_{\text{encoder},0}))$$

where $\mathbf{H}_{\text{encoder},0}$ and $\mathbf{C}_{\text{encoder},0}$ are the initial hidden and cell states, respectively, and $\mathbf{H}_{\text{encoder}}$ and $\mathbf{C}_{\text{encoder}}$ are the final hidden and cell states.

3. *Decoder LSTM*: For the target sequence $\mathbf{Y} \in \mathbb{R}^{\text{batch size} \times \text{sequence length}}$, the operation performed by the decoder LSTM is:

$$\mathbf{H}_{\text{decoder}} = \text{LSTM}_{\text{decoder}}(\mathbf{E}, (\mathbf{H}_{\text{encoder}}, \mathbf{C}_{\text{encoder}}))$$

where $\mathbf{E} = \text{Embedding}(\mathbf{Y})$ is the embedded target sequence.

4. *Fully Connected Layer*: The output of the model is:

$$\mathbf{O} = \mathbf{H}_{\text{decoder}} \cdot \mathbf{W}_{fc} + \mathbf{b}_{fc}$$

where $\mathbf{W}_{fc} \in \mathbb{R}^{\text{hidden size} \times \text{output size}}$ are the weights of the fully connected layer, and $\mathbf{b}_{fc} \in \mathbb{R}^{\text{output size}}$ is the bias.

5 Experiments

5.1 Preprocessing

To facilitate analysis, the raw Mel-frequency cepstral coefficients (MFCC) and chroma features were extracted from the audio signal to serve as input for the model. Utilizing the audio spectrum better portrays sequential dependencies and patterns, surpassing the capabilities of analyzing the raw audio signal alone. The corresponding text data was transformed into numerical tokens to be used by the model. Finally, the audio features and tokenized text were mapped together as input for the model. The train-validation-test split scheme was 70-15-15.

5.2 Procedure

The models used for training were RNN, LSTM, GRU, CNN-RNN, CNN-GRU, and CLSTMA-MT. Each model was trained using a CrossEntropyLoss criterion with a learning rate of 0.001, an Adam optimizer, and a dropout rate 0.2. The batch size of training, validation, and testing were in batches of six 256. The training was performed over 200 epochs for all models. The training loss was computed for each batch, and the model parameters were updated to minimize the loss. After an epoch of training, the model's performance was evaluated on a validation dataset. After all epochs were completed, the final model was tested on a testing dataset to compute the testing loss.

5.3 Inference

Inferences could be performed to generate a sequence of descriptions given input audio. Each model can take an audio file as input, extract features from it, and pass these features through the model to generate a sequence of word tokens. These tokens are then decoded back into words to get the predicted text description of the music.

6 Evaluation and Results

Our music tagging task is a multi-label classification task since each audio file is associated with multiple aspects or attributes. Traditional metrics like precision, recall, and F1 score may not be entirely suitable for evaluating the model performance due to the variability and complexity of the labels. We use the cross-entropy test loss to evaluate each model's reconstruction capabilities, as it quantifies the difference between the predicted and target distributions. All hybrid CNN implementations achieve the lowest loss on average.

Model	RNN	GRU	LSTM	CNN_RNN	CNN_GRU	ALSTM-MT
Loss	4.08	4.42	4.30	3.80	3.77	**2.89**

Given the subjective nature of the data, a qualitative assessment might be advantageous. This would entail examining the model's output in light of human perception and interpretation. In this context, we sought the opinion of a professional musician. They observed that while the model effectively identified the instrumentation in the songs, it missed grasping the comprehensive structure of the music. The model focused only on specific cues that provided insight into less significant elements of the song, offering a superficial understanding of the music. Given this, the musician recommended reconsidering the detailed descriptions, suggesting they could be misleading.

7 Discussion

The superior performance of CNNs can be attributed to their alignment with the neurophysiological characteristics of the human auditory system [8]. The mel-spectrogram, which serves as the input representation, mimics how humans perceive sound by emphasizing frequency bands crucial for discrimination. By processing sound in a spectrotemporal manner, CNNs learn filters that resemble the spectrotemporal receptive field (STRF) observed in the brain. This transformation allows CNNs to generate a linearly separable representation from the original mel-spectrogram [14]. Consequently, CNNs excel at capturing local features and spatial relationships, enabling them to process spectrotemporal patterns in audio signals effectively. On the other hand, RNNs are more suitable for handling sequential data and may not perform as proficiently as CNNs in this context.

Interestingly, the RNN-only model achieved the lowest loss despite its relative simplicity compared to its variants GRU and LSTM, which are better at capturing long-term dependencies. This advantage is likely to come from the nature of the data itself. Each music audio sample is 20 seconds long, a relatively short time for a song. This means that time-dependent features in the music data don't have long-term dependencies. In other words, the current output depends mainly on the recent past rather than on information from far back in the sequence. This may explain why the RNN could perform just as well, if not better, than more complex models like GRUs and LSTMs.

CLSTMA-MT achieving the lowest loss is likely due to the complementary nature of CNNs and LSTMs when handling sequential data. The strength of CNNs in processing these spectrotemporal patterns might account for their superior performance when paired with an LSTM, which would help capture temporal dependencies. Despite its sophistication, the model might not necessarily outperform simpler models due to the complexity and variability of the task, as well as the potential for overfitting with a more complex model.

It's also worth noting that the performance of a model can be affected by the specifics of the training process, including the number of epochs and the batch size, as well as the optimizer and learning rate used. The settings we used were primarily default ones, and more experimentation is required to understand the overall performance better.

Lastly, we originally planned to use MuLAN as a benchmark, but we faced several issues that could have improved our progress. MuLAN is a joint audio-text embedding model that connects music audio with natural language music descriptions [15]. It utilizes separate audio and text embedding networks and is trained on a large dataset of music recordings and weakly-associated text annotations. The model aims to minimize a Contrastive Multiview Coding loss by maximizing the similarity between target audio-text pairs and minimizing the similarity between non-target pairs within each batch.

However, some of the main functions to implement MuLAN were undefined in the code base, so the implementation could never come to fruition [16]. Without MuLAN as a reference, we utilized cross-entropy loss instead of contrastive loss. Since music-text pairs are different modalities, the latter loss function may have also provided promising results, and its implementation is a potential research direction for automatic music tagging.

8 Conclusion

In this project, we proposed the CLSTMA-MT for generating detailed music captions based on the essence of music recordings. We adopted an autoencoder structure, integrating LSTM and CNN components, and trained the model on the MusicCaps dataset. We evaluated the performance of different baseline models and found that the hybrid CNN-LSTM model achieved the lowest cross-entropy loss. The versatility of our system enables its application in various music-related domains, such as audio-based music retrieval, recommendation, playlist generation, and analysis. However, evaluating the model's output should be considered qualitatively, considering human perception and interpretation. The performance of CNNs can be attributed to their alignment with the neurophysiological characteristics of the human auditory system, while RNNs are more suitable for processing sequential data. Further investigation is needed to explore different training settings and the potential of using contrastive loss in the future.

9 References

[1] A. Ramesh *et al*., "Zero-Shot Text-to-Image Generation." arXiv, Feb. 26, 2021. Accessed: May 29, 2023. [Online]. Available: http://arxiv.org/abs/2102.12092

[2] A. Agostinelli *et al*., "MusicLM: Generating Music From Text." arXiv, Jan. 26, 2023. doi: 10.48550/arXiv.2301.11325.

[3] "MusicCaps." https://www.kaggle.com/datasets/googleai/musiccaps.

[4] G. Song, Z. Wang, F. Han, S. Ding, and M. A. Iqbal, "Music auto-tagging using deep Recurrent Neural Networks," *Neurocomputing*, vol. 292, pp. 104–110, May 2018, doi: 10.1016/j.neucom.2018.02.076.

[5] C. Kakarla, V. Eshwarappa, L. Babu Saheer, and M. Maktabdar Oghaz, "Recurrent Neural Networks for Music Genre Classification," in *Artificial Intelligence XXXIX*, M. Bramer and F. Stahl, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2022, pp. 267–279. doi: 10.1007/978-3-031-21441-7_19.

[6] M. Won, A. Ferraro, D. Bogdanov, and X. Serra, "Evaluation of CNN-based Automatic Music Tagging Models." arXiv, Jun. 01, 2020. Accessed: Mar. 08, 2023. [Online]. Available: http://arxiv.org/abs/2006.00751

[7] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The handbook of brain theory and neural networks*, Cambridge, MA, USA: MIT Press, 1998, pp. 255–258.

[8] K. Choi, G. Fazekas, and M. Sandler, "Automatic tagging using deep convolutional neural networks." arXiv, Jun. 01, 2016. Accessed: May 29, 2023. [Online]. Available: http://arxiv.org/abs/1606.00298

[9] S. Dieleman and B. Schrauwen, "MULTISCALE APPROACHES TO MUSIC AUDIO FEATURE LEARNING".

[10] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, "Detection and Classification of Acoustic Scenes and Events," *IEEE Trans. Multimedia*, vol. 17, no. 10, pp. 1733–1746, Oct. 2015, doi: 10.1109/TMM.2015.2428998.

[11] J. D. da Silva, Y. M. G. da Costa, and M. A. Domingues, "Improving Musical Tag Annotation with Stacking and Convolutional Neural Networks," in *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, Jul. 2020, pp. 393–398. doi: 10.1109/IWSSIP48289.2020.9145192.

[12] S. Hershey *et al*., "CNN Architectures for Large-Scale Audio Classification." arXiv, Jan. 10, 2017. Accessed: Mar. 09, 2023. [Online]. Available: http://arxiv.org/abs/1609.09430

[13] K. Drossos, S. Adavanne, and T. Virtanen, "Automated Audio Captioning with Recurrent Neural Networks." arXiv, Oct. 24, 2017. Accessed: May 29, 2023. [Online]. Available: http://arxiv.org/abs/1706.10006

[14] M. Dong, "Convolutional Neural Network Achieves Human-level Accuracy in Music Genre Classification." arXiv, Feb. 26, 2018. Accessed: May 29, 2023. [Online]. Available: http://arxiv.org/abs/1802.09697

[15] Q. Huang, A. Jansen, J. Lee, R. Ganti, J. Y. Li, and D. P. W. Ellis, "MuLan: A Joint Embedding of Music Audio and Natural Language." arXiv, Aug. 25, 2022. Accessed: Mar. 07, 2023. [Online]. Available: http://arxiv.org/abs/2208.12415

[16] "audio_lm and mulan_embed_quantizer are not defined · Issue #52 · lucidrains/musiclm-pytorch," *GitHub*. https://github.com/lucidrains/musiclm-pytorch/issues/52 (accessed May 29, 2023).

10 Contributions

Kevin ZHANG • 50004513

- Proposed the project topic
- Wrote the proposals and final report
- Compiled and shared relevant literature for an understanding of the overall methodology
- Provided qualitative feedback on the final predictions
- Created slides, recorded, and edited the video presentation

Zusheng TAN • 20992699

- Designed the model architectures
- Designed the pipelines for preprocessing, training, validation, testing, and inference
- Ran experiments for each model with various parameter settings
- Compiled and shared relevant literature for the modeling phase
- Recorded for the video presentation

Hanyu DUAN • 20652756

- Worked on implementing MuLAN to use as a benchmark