Analysis of Weighted BCE Loss Functions for Music Tagging

Yifeng Yu, Alexander Lerch
Center for Music Technology, Georgia Institute of Technology, USA
{yyu479,alexander.lerch}@gatech.edu

***Abstract*—In this paper, I reproduce and evaluate several machine learning models including SVM, FCN, CRNN, and short-chunk CNN for music tagging and explore the impact of using different weighted binary cross-entropy (BCE) loss functions. The models are trained on the MagnaTagATune dataset with 50 labels. I evaluate the models using multiple metrics, including ROC-AUC, PR-AUC, precision, recall, and F1 score.**

I. Introduction

THE expansion of online digital music streaming platforms has led to increasing demand for efficient and effective ways to organize and search large music libraries. Music tagging aims to address this challenge by assigning descriptive labels or tags to pieces of music based on their audio features. These tags can help users discover new music that matches their preferences, build personalized playlists, and navigate music collections more easily.

In this paper, we focus on reproducing and evaluating music tagging models, including SVM and other deep learning models. We explore the impact of using different weighted binary cross-entropy (BCE) loss functions and evaluate the models using multiple metrics, including ROC-AUC, PR-AUC, precision, recall, and F1 score. Additionally, we investigate the performance of these models on inputs of varying lengths, to understand their behavior in different contexts.

Our contributions in this paper include reproducing and evaluating existing deep learning models for music tagging, investigating the impact of using different loss functions, and evaluating different models. Through our experiments, we aim to provide insights into the behavior of different models for music tagging, and help inform future research in this field.

II. Related Work

The task of music tagging has been approached using various techniques. Previous studies have explored the application of machine learning and deep learning techniques for music tagging. Machine learning models including supervised learning models such as Support Vector Machines (SVMs) [1], Linear Regression [2] and unsupervised learning models such as K-means [3], hidden Markov model [4] have been applied to this task. However, with the increasing availability of large annotated datasets and advancements in deep learning, many recent studies have focused on deep learning-based approaches. In particular, Convolutional Neural Networks (CNNs) have been widely used for music tagging tasks [5]–[10]. These models can automatically learn hierarchical representations of audio signals and have achieved state-of-the-art performance in various music classification tasks. In addition to CNNs, Recurrent Neural Networks (RNNs) [11] as well as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) [12] have also been used for music tagging. These models can capture temporal dependencies in audio signals and are particularly effective in modeling sequential data such as music. Besides, Convolutional Recurrent Neural Network (CRNN) [13] takes advantage of both CNN and RNN. Furthermore, Transformer-based models have recently shown promising results in various natural language processing tasks and have also been applied to music tagging [14]–[19]. These models can learn global dependencies in the input sequence and have the potential to capture long-range dependencies in music signals. Moreover, the choice of loss functions can have a significant impact on the performance of deep learning models. For example, Greer [20] presents an ensemble-based CNN model trained using various loss functions to tag musical genres from audio.

III. Method

In this study, we explore the effectiveness of a traditional machine learning method and deep learning methods for multi-label music tagging. For the traditional machine learning method, we use support vector machine (SVM) as the baseline to perform binary classification between two labels. Specifically, we train 50 SVM models for each label combination, and evaluate the performance of the SVMs using the mean F1 score for all labels.

For the deep learning methods, we experiment with three different architectures: Fully Convolutional Network (FCN) [5], CRNN [13], and short-chunk CNN [21]. We use log-mel spectrogram as the input feature. For FCN and CRNN, we use 30 seconds as the input length, for short-chunk CNN which is designed for short-chunk input, we use both 10 seconds and 30 seconds to test if the input length will affect the performance.

In the dataset, 1 represents included labels whereas 0 represents excluded labels. We found that the average number of included labels for each sample is 2.68. Figure 1 shows the distribution of the labels, which is heavily imbalanced. Ordinary BCE loss, on the other hand, assigns equal weight to both cases with and without labels, which may result in the model predicting all values to be 0. Therefore, to address the issue of imbalanced labels, we experiment with different loss functions, including binary cross-entropy (BCE) loss and weighted BCE loss. The weighted BCE loss function assigns

different weights to included and excluded labels based on their frequency. We consider BCE loss and four different settings for the weighted BCE loss function:

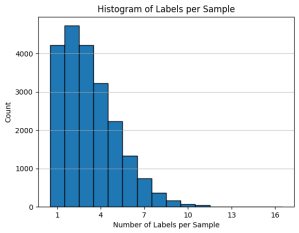

Fig. 1. Included labels per sample distribution

1) BCE loss:

$$L(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}\Big[y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)\Big]$$

In this formula, $L(y, \hat{y})$ represents the BCE loss for true labels y and predicted labels $\hat{y}$, where y_i and $\hat{y}_i$ are the true and predicted labels for the i^{th} sample, respectively. N is the total number of samples.

2) Global class weighted BCE loss:

$$w_c = 1 - \frac{N_i}{N}$$
$$L_{\text{global class}}(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}\Big[w_{c_i} \cdot y_i \cdot \log(\hat{y}_i) + w_{c_{(1-y_i)}} \cdot (1 - y_i) \cdot \log(1 - \hat{y}_i)\Big] \quad (1)$$

In this formula, $L_{\text{global class}}(y, \hat{y})$ represents the global class weighted BCE loss. w_c denotes the weight for class c, where N is the total number of samples, and N_i is number of samples in class c. c_i represents the class of the i^{th} sample. This loss function is designed to take into account the global-level class distributions, which can lead to better performance on imbalanced classes.

3) Batch-level included/excluded weighted BCE loss:

$$w_{\text{batch inc}} = 1 - \frac{P_B}{T_B}$$
$$w_{\text{batch exc}} = 1 - w_{\text{batch inc}}$$
$$L_{\text{batch inc/exc}}(y, \hat{y}) = -\frac{1}{N}\sum_{j=1}^{M}\sum_{i=1}^{B}\Big[w_{\text{batch inc}} \cdot y_{ij} \cdot \log(\hat{y}_{ij}) + w_{\text{batch exc}} \cdot (1 - y_{ij}) \cdot \log(1 - \hat{y}_{ij})\Big] \quad (2)$$

In this formula, $L_{\text{batch inc/exc}}(y, \hat{y})$ represents the batch-level included/excluded weighted BCE loss. y_{ij} and $\hat{y}_{ij}$ are the true and predicted labels for the i^{th} sample within the j^{th} batch, respectively. B is the batch size, M is the total number of batches in an epoch, and $w_{\text{batch inc}}$ and $w_{\text{batch exc}}$ are the weights for included and excluded labels that are calculated based on the total labels (T_B) and the number of included labels (P_B) in the batch, respectively. This loss function is designed to balance the weights when the data has imbalanced 1 and 0 labels.

4) Global-level included/excluded weighted BCE loss:

$$w_{\text{global inc}} = 1 - \frac{P}{T}$$
$$w_{\text{global exc}} = 1 - w_{\text{global inc}}$$
$$L_{\text{global inc/exc}}(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}\Big[w_{\text{global inc}} \cdot y_i \cdot \log(\hat{y}_i) + w_{\text{global exc}} \cdot (1 - y_i) \cdot \log(1 - \hat{y}_i)\Big] \quad (3)$$

In this formula, $L_{\text{batch inc/exc}}(y, \hat{y})$ represents the global-level included/excluded weighted BCE loss where $w_{\text{global inc}}$ and $w_{\text{global exc}}$ are the weights for included and excluded labels at the global level, respectively. The weights $w_{\text{global inc}}$ and $w_{\text{global exc}}$ are calculated based on the total labels (T) and the number of included labels (P) across all data.

5) Combined weighted BCE loss:

$$L_{\text{combined}}(y, \hat{y}) = -\frac{1}{N}\sum_{j=1}^{M}\sum_{i=1}^{B}\Big[w_{c_i} \cdot w_{\text{batch inc}} \cdot y_{ij} \cdot \log(\hat{y}_{ij}) + w_{c_i} \cdot w_{\text{batch exc}} \cdot (1 - y_{ij}) \cdot \log(1 - \hat{y}_{ij})\Big] \quad (4)$$

In this formula, $L_{\text{combined}}(y, \hat{y})$ represents combined weights BCE loss, where the weights are obtained by multiplying the global class weights with the batch-level included/excluded weights.

IV. Experiments

A. Datasets

In this section, we present our experiments for multi-label music tagging using the MagnaTagATune dataset [22]. The MagnaTagATune dataset contains 25,877 audio clips, each with a duration of 29.1 seconds on average, and is annotated with 188 labels. We use the same data split as in previous studies, with the first 12 folders (0-b) for training, the 13th (c) for validation, and the last three folders (d-e) for testing.

B. Preprocessing

Before extracting the features, we first preprocessed the labels by selecting the top 50 labels based on the frequency and merging synonyms. For example, we merged labels such as “beat” and “beats”, “female” and “women”, and so on. Figure 2 shows the distribution of the original labels in the MagnaTagATune dataset, while Figure 3 shows the distribution of the preprocessed labels. As can be seen from the figures, the preprocessing step resulted in a significant reduction in the number of labels, from 188 to 50. The preprocessed labels are also more evenly distributed than the original labels, which could help us focus on the most relevant and informative labels for music tagging and improve the performance of our models.

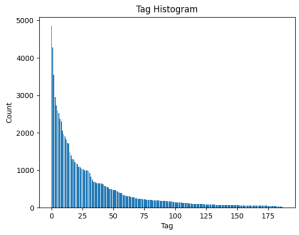

Fig. 2. Original label distribution

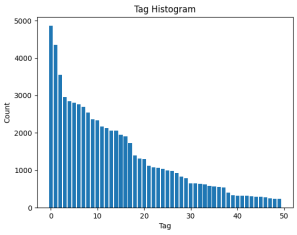

Fig. 3. Preprocessed label distribution

C. Feature extraction

We took raw WAV files with a sample rate of 16000 as input, and extracted MFCC of 25 dimensions and log-mel spectrogram of 128 dimensions. For the SVM, we used MFCC and log-mel spectrogram with and without PCA. For the deep learning models, we used log-mel spectrogram as input, with $n_{fft} = 512$, $f_{min} = 0$, $f_{max} = 8000$, and a hop size of 256. The original audio length was 29.1 seconds, which was padded to 30 seconds. For models like short-chunk CNN, which are designed for short-chunk inputs, we cropped the audio into segments. Specifically, we used 10 seconds as the input length and 5 seconds as the shift length, resulting in 5 segments for each audio sample. We used the same labels as the 30-second audio sample for all segments.

D. Experimental setup

To train the SVM, we developed 50 binary classifiers for each label using the mean value of MFCC and log-mel spectrogram, respectively. We used MFCC with and without principal component analysis (PCA), reducing the feature dimension to 10 components. These input features were used for both 10-second and 30-second durations. The evaluation metric used for these classifiers was the F1 score. We also employed the RandomUnderSampler class from the imblearn package in Python to achieve a balance between the number of 1 and 0 labels and to investigate whether the issue of imbalanced data would impact the model's performance.

For the deep learning models, we used log-mel spectrogram and three CNN-based architectures: FCN, CRNN, and short-chunk CNN, and tested different loss functions as mentioned before. We also experimented with different input lengths, loss functions, and model architectures. We trained our models using Pytorch, with a learning rate of 0.0001, batch size of 16, and AdamW optimizer with weight decay of 0.0001. We also used a dropout rate of 0.5 to prevent overfitting.

V. Evaluation

Due to the imbalanced data labels mentioned before, we cannot accurately measure the performance of the model with a simple accuracy rate. Therefore, we decided to use PR AUC, ROC AUC, precision, recall, and F1 score to measure the performance of our models. Since our SVM models are binary, we evaluated the F1 score for all 50 models and calculated the average to obtain a more robust measure of overall performance. PR AUC and ROC AUC are commonly used metrics for evaluating multi-label classification tasks, which take into account both precision and recall across different decision thresholds. Precision is the fraction of relevant instances among the total number of predicted positive instances, while recall is the fraction of relevant instances among the total number of actual positive instances. F1 score is the harmonic mean of precision and recall, and provides a single value that balances both metrics. By using these metrics, we can evaluate the models from different aspects and obtain a more accurate assessment of their performance.

In addition to the standard evaluation metrics, we also performed an analysis to identify the model performance of each label. We examined the confusion matrices for each label and analyzed the least and the most frequently misclassified labels. This allowed us to gain insights into the strengths and weaknesses of our models and identify areas for improvement.

VI. Results and Discussion

A. SVM Results

Based on our experiments using 25-dimensional MFCC features and 128-dimensional log mel-spectrogram with and without PCA, as well as 10s and 30s audio clips as inputs, we have found the following:

- Log mel-spectrogram features perform better than MFCC features.
- Using 30s audio clips as input yields better results than using 10s clips.
- When dealing with imbalanced data, the F1 score is close to 0 and the model outputs almost all 0s.

Our experimental results are summarized in Table I, which shows that the combination of log mel-spectrogram features and 30s audio clips achieved the highest F1 score.

Input Feature	Duration	Data Balance	Avg F1-score
PCA(10) MFCC	10s	Balanced	0.1797
PCA(10) MFCC	30s	Balanced	0.1881
MFCC	10s	Balanced	0.2041
MFCC	30s	Balanced	0.2097
MFCC	30s	Unbalanced	0.0263
log mel	10s	Balanced	0.2181
log mel	30s	Balanced	**0.2198**

TABLE I
SVM EXPERIMENT RESULTS WITH DIFFERENT CONFIGURATIONS

B. DNN Results

Table II shows the performance of different deep learning models with different input duration and losses.

Our experimental results indicate the following:

- Short-Chunk CNN model performs the best among the three models.
- Using a longer chunk size of 30 seconds yielded better results than using a chunk size of 10 seconds. This suggests that incorporating more context into the model can improve its ability to accurately tag music, which is consistent with the result of the SVM model.
- Using the included/excluded weighted BCE loss functions resulted in improved recall and ROC-AUC in the

Model	Duration	Loss	Accuracy	Best Threshold	Precision	Recall	F1	ROC AUC	PR AUC
FCN	30s	BCE	0.83	0.051	0.219	0.871	0.35	0.922	0.483
FCN	30s	WBCE global cls	0.835	0.055	0.223	0.864	0.355	0.922	0.474
FCN	30s	WBCE batch i/e	0.825	0.384	0.216	0.883	0.347	0.925	0.467
FCN	30s	WBCE global i/e	0.833	0.418	0.223	0.875	0.355	0.925	0.467
FCN	30s	WBCE combined	0.833	0.434	0.223	0.873	0.355	0.925	0.465
CRNN	30s	BCE	0.821	0.052	0.204	0.832	0.328	0.898	0.397
CRNN	30s	WBCE global cls	0.813	0.051	0.198	0.832	0.319	0.898	0.392
CRNN	30s	WBCE batch i/e	0.826	0.476	0.215	0.876	0.346	0.919	0.44
CRNN	30s	WBCE global i/e	0.824	0.473	0.213	0.869	0.342	0.917	0.426
CRNN	30s	WBCE combined	0.819	0.061	0.204	0.84	0.328	0.903	0.398
short-chunk CNN	10s	BCE	0.845	0.055	0.236	0.866	0.371	0.929	0.501
short-chunk CNN	10s	WBCE global cls	0.828	0.048	0.219	0.883	0.351	0.927	0.496
short-chunk CNN	10s	WBCE batch i/e	0.836	0.477	0.227	0.879	0.36	0.929	0.486
short-chunk CNN	10s	WBCE global i/e	0.849	0.493	0.241	0.872	0.377	0.931	0.491
short-chunk CNN	10s	WBCE combined	0.834	0.443	0.225	0.881	0.358	0.929	0.485
short-chunk CNN	30s	BCE	**0.851**	0.055	**0.245**	0.875	**0.382**	**0.933**	**0.506**
short-chunk CNN	30s	WBCE global cls	0.814	0.046	0.202	0.857	0.327	0.909	0.428
short-chunk CNN	30s	WBCE batch i/e	0.812	0.468	0.201	0.867	0.327	0.914	0.425
short-chunk CNN	30s	WBCE global i/e	0.831	0.463	0.223	**0.890**	0.357	0.930	0.476
short-chunk CNN	30s	WBCE combined	0.834	0.475	0.226	0.885	0.360	0.931	0.486

TABLE II
DEEP LEARNING MODELS PERFORMANCE METRICS

Ranking	Label	Count	SVM		FCN		CRNN		short-chunk CNN	
			Label	F1	Label	F1	Label	F1	Label	F1
1	guitar	4861	rock	0.58	opera	0.60	rock	0.61	opera	0.66
2	classical	4358	opera	0.46	rock	0.60	techno	0.55	choir	0.62
3	slow	3547	classical	0.45	heavy	0.55	opera	0.53	rock	0.62
4	techno	2954	techno	0.42	techno	0.51	heavy	0.44	heavy	0.56
5	string	2842	guitar	0.42	ambient	0.47	classical	0.43	techno	0.55
6	vocal	2813	choir	0.42	choir	0.47	guitar	0.40	female	0.53
7	electro	2764	piano	0.40	harpsichord	0.46	choir	0.40	ambient	0.51
8	drum	2698	slow	0.40	classical	0.44	beat	0.40	harpsichord	0.48
9	no singer	2550	harpsichord	0.37	female	0.43	piano	0.38	classical	0.46
10	rock	2371	heavy	0.36	guitar	0.41	hard	0.38	guitar	0.45
					...					
41	bass/modern	327	trance	0.06	eastern	0.14	jazz	0.03	trance	0.20
42	no piano	324	orchestra	0.05	trance	0.13	baroque	0	cello	0.16
43	hard	323	cello	0.05	cello	0.13	bass	0	strange	0.16
44	chant	312	foreign	0.05	foreign	0.12	eastern	0	folk	0.13
45	baroque	297	bass	0.04	strange	0.12	folk	0	eastern	0.13
46	orchestra	296	no beat	0.04	bass	0.10	foreign	0	foreign	0.12
47	foreign	275	no piano	0.04	folk	0.04	modern	0	bass	0.10
48	trance	253	folk	0.03	modern	0.02	no beat	0	modern	0.08
49	folk	243	modern	0.03	no beat	0	no piano	0	no beat	0.03
50	no beat	242	eastern	0.02	no piano	0	trance	0	no piano	0

TABLE III
TOP 10 AND BOTTOM 10 F1 SCORES FOR ALL 50 LABELS

first three settings, although it reduced the PR AUC compared to using the standard BCE loss function. Furthermore, the use of the included/excluded weighted BCE loss function can also improve the threshold selection process, leading to a more centralized threshold. However, we found in the last setting where we use short-chunk CNN with 30-second durations, the weighted BCE doesn't outperform BCE loss.

C. Compare Results Between Labels

Table III provides a list of the F1 scores for the top 10 and bottom 10 labels sorted in descending order. For the sake of consistency, we have uniformly adopted a 30-second audio length and BCE loss for all models as the baseline to compare.

We observed that the F1 score for each label was highly related to its total number of occurrences in the dataset. Labels that appeared more frequently tended to have higher F1 scores, indicating the challenge of imbalanced multi-label datasets in music tagging.

VII. CONCLUSION

In conclusion, our study compares the performance of four different models, including an SVM baseline, and investigates the impact of five different loss functions on music tagging. Our experimental results show that using short-chunk CNNs outperforms other models, and using included/excluded weighted BCE loss function improves the recall and ROC-AUC of most models. Moreover, we observe that the F1 scores are positively correlated with the total number of the corresponding labels in the dataset, highlighting the challenge of dealing with label imbalance in music tagging. Our findings suggest that ensemble-based deep learning models trained with proper loss functions are effective for music tagging, and future research can explore techniques to better address label imbalance and improve the interpretability of the models.

REFERENCES

[1] A. Kumar and B. Raj, "Audio event detection using weakly labeled data," in *Proceedings of the 24th ACM international conference on Multimedia*, 2016, pp. 1038–1047.

[2] A. Van Den Oord, S. Dieleman, and B. Schrauwen, "Transfer learning by supervised pre-training for audio-based music classification," in *Conference of the International Society for Music Information Retrieval (ISMIR 2014)*, 2014.

[3] G. Chen and B. Han, "Improve k-means clustering for audio data by exploring a reasonable sampling rate," in *2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery*, vol. 4. IEEE, 2010, pp. 1639–1642.

[4] X. Shao, C. Xu, and M. S. Kankanhalli, "Unsupervised classification of music genre using hidden markov model," in *2004 IEEE International Conference on Multimedia and Expo (ICME)(IEEE Cat. No. 04TH8763)*, vol. 3. IEEE, 2004, pp. 2023–2026.

[5] K. Choi, G. Fazekas, and M. Sandler, "Automatic tagging using deep convolutional neural networks," *arXiv preprint arXiv:1606.00298*, 2016.

[6] J. Pons, O. Nieto, M. Prockup, E. Schmidt, A. Ehmann, and X. Serra, "End-to-end learning for music audio tagging at scale," *arXiv preprint arXiv:1711.02520*, 2017.

[7] J. Lee, J. Park, K. L. Kim, and J. Nam, "Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms," *arXiv preprint arXiv:1703.01789*, 2017.

[8] T. Kim, J. Lee, and J. Nam, "Sample-level cnn architectures for music auto-tagging using raw waveforms," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2018, pp. 366–370.

[9] M. Won, S. Chun, O. Nieto Caballero, and X. Serra, "Automatic music tagging with harmonic cnn," 2019.

[10] M. Won, S. Chun, O. Nieto, and X. Serrc, "Data-driven harmonic filters for audio representation learning," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 536–540.

[11] 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, 2018.

[12] T. Cai, M. I. Mandel, and D. He, "Music autotagging as captioning," in *Proceedings of the 1st Workshop on NLP for Music and Audio (NLP4MusA)*, 2020, pp. 67–72.

[13] K. 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)*. IEEE, 2017, pp. 2392–2396.

[14] M. Won, S. Chun, and X. Serra, "Toward interpretable music tagging with self-attention," *arXiv preprint arXiv:1906.04972*, 2019.

[15] M. Sukhavasi and S. Adapa, "Music theme recognition using cnn and self-attention," *arXiv preprint arXiv:1911.07041*, 2019.

[16] G. Song, Z. Wang, F. Han, S. Ding, and X. Gu, "Music auto-tagging using scattering transform and convolutional neural network with self-attention," *Applied Soft Computing*, vol. 96, p. 106702, 2020.

[17] Y. Yu, S. Luo, S. Liu, H. Qiao, Y. Liu, and L. Feng, "Deep attention based music genre classification," *Neurocomputing*, vol. 372, pp. 84–91, 2020.

[18] M. Won, K. Choi, and X. Serra, "Semi-supervised music tagging transformer," *arXiv preprint arXiv:2111.13457*, 2021.

[19] C. Ju, L. Han, and G. Peng, "Music auto-tagging based on attention mechanism and multi-label classification," in *The International Conference on Image, Vision and Intelligent Systems (ICIVIS 2021)*. Springer, 2022, pp. 245–255.

[20] T. Greer, X. Shi, B. Ma, and S. Narayanan, "Multi-modal, multi-task, music bert: A context-aware music encoder based on transformers," 2022.

[21] M. Won, A. Ferraro, D. Bogdanov, and X. Serra, "Evaluation of cnn-based automatic music tagging models," *arXiv preprint arXiv:2006.00751*, 2020.

[22] E. Law, K. West, M. I. Mandel, M. Bay, and J. S. Downie, "Evaluation of algorithms using games: The case of music tagging." in *ISMIR*. Citeseer, 2009, pp. 387–392.