AN AUTO-TAGGING BASED APPROACH FOR APPROXIMATING ECHONEST MUSICAL FEATURES

Jason Thomo
University of Victoria
jthomo@uvic.ca

Davy Bossin
University of Victoria
davybossin@uvic.ca

Daniel Chrenko
University of Victoria
danielchrenko@uvic.ca

ABSTRACT

The acquisition of The Echo Nest by Spotify in 2016 marked a significant shift in the landscape of music analysis tools. Previously an open-source beacon in the music technology community, The Echo Nest's algorithm was transformed into a proprietary, closed-source "black box," accessible only through Spotify's Web API. This change severely restricted transparency and hindered the ability of independent developers, researchers, and music enthusiasts to access detailed music analysis and feature extraction tools that were once readily available. In response to this challenge, we develop an open-source alternative that offers a higher level of simplicity and accessibility. By creating a multiclass classifier model, we aim to approximate the numeric scores of The Echo Nest with descriptive tags for any given audio track. Our model can also give insight into which musical features provide the strongest associations with Echo Nest scores. Our approach emphasizes interpretability and user-friendliness, making it suitable for a wide range of users from academic researchers to hobbyist music curators. The simplicity of our model makes it easy to be used in practical applications like music discovery, categorization, and recommendation.

1. INTRODUCTION

The 2016 acquisition of The Echo Nest by Spotify transformed its previously open-source algorithm into a proprietary, closed-source "black box," accessible solely through Spotify's Web API. This shift reduced transparency in how the tool's music analysis and feature extraction works, leaving a gap for users who rely on this data for academic and practical applications. We addressed this challenge by developing an open-source multiclass classifier model designed to approximate The Echo Nest's numeric scores with descriptive tags for given audio tracks. This model simplifies the music analysis process and provides more insight into which musical features have strong associations with EchoNest scores.

Our classification model approximates EchoNest scores for given audio tracks via auto-tagging. The model outputs tags corresponding to the presence of each feature in a given audio track based on predefined thresholds. For EchoNest scores such as "Danceability", "Acousticness", and "Energy", our model will generate corresponding tags "Danceable", "Acoustic", and "Energetic", respectively. These tags facilitate the efficient categorization and retrieval of music based on a variety of well-known audio characteristics.

For the preparation and preprocessing of audio data, we use the Free Music Archive (FMA) dataset [1]. This dataset provides a diverse collection of songs (sound files) with associated metadata. Our preprocessing steps involve labeling audio tracks with tags if they meet or exceed calculated thresholds for each EchoNest score.

We use a supervised learning approach, with the audio tracks serving as inputs and the tags as outputs. Following a similar methodology to that outlined in "Semantic Annotation and Retrieval of Music and Sound Effects," [2] we represent each audio track as a bag of feature vectors. Each vector captures essential characteristics of the music, such as tempo, Mel-Frequency Cepstral Coefficient (MFCC), Chroma content, among others, which are generated using the Librosa Python library [3]. This representation enables our model to learn associations between Librosa-generated musical features and the target EchoNest tags.

In the training and evaluation of the model, we optimize the model's ability to associate specific audio features with tags by testing different classifier models, varying tag/label generation thresholds, varying the input feature selection threshold based on feature importance, as well as expanding the training dataset to include more audio tracks from which the model can learn.

The final model is capable of taking an audio track as input and outputting tags that describe its characteristics similar to EchoNest scores. This auto-tagging system provides a valuable tool for music discovery and recommendation platforms, enabling efficient categorization and retrieval of music based on a wide range of audio attributes.

2. TOOLS AND RESOURCES

FMA dataset: The FMA (Free Music Archive) dataset consists of sound files and their associated metadata [1]. The "FMA Metadata" subdirectory of the dataset contains a file "echonest.csv" which contains a list of approximately 13,000 associations between song files and EchoNest scores. Our final model uses a subset of these files which are also contained in the "FMA Medium" subdirectory, totalling approximately 5,300 files.

EchoNest features: EchoNest features are numeric scores given to audio tracks across various categories. We utilize the following features: "Acousticness", "Danceability", "Energy", "Instrumentalness", "Liveliness", "Speechiness" and "Valence". The FMA dataset provides a set of tracks which have these features pre-computed and stored in the "echonest.csv" file in the "FMA_metadata" subdirectory, so we do not need a separate API to generate these features.

Librosa: Librosa [3] is a Python package that is widely used to process and extract musical features from audio files. We use Librosa to generate features such as MFCC, Tempo, and Chroma from sound files to feed to our model.

Scikit-learn: Scikit-learn [4] is a Python package that provides a variety of classic machine learning models. We evaluate various classifier models such as Random Forest, Support Vector Machine (SVM), Logistical Regression, and Multi-layer Perceptron (MLP) to determine which are most effective for our multi-class classification problem.

3. METHODOLOGY

3.1 Audio dataset preprocessing

We adapt the FMA Medium dataset of sound files, originally containing 30-second audio clips, to enhance processing efficiency. We take only the first 6 seconds of each track, significantly reducing computational demands while still maintaining enough information to make effective predictions. To streamline the retrieval of audio tracks, we also categorize them by genre based on the metadata provided in the FMA dataset. This organization may enable the future iterations of the model to learn and predict based on genre-specific audio characteristics.

3.2 EchoNest score normalization

In the process of preparing the data for our machine learning model, we observed that several of the numeric features provided by EchoNest have highly skewed distributions. This skewness can be problematic for many machine learning algorithms, as they often assume that the input features follow a more symmetric, Gaussian-like distribution. To address this issue, we apply a Gaussian scaling normalization technique using Scikit-learn's PowerTransformer to all the original EchoNest features. By applying this transformation, we obtain a more balanced and compressed distribution. We also experimented with unit range scaling using Scikit-learn's MinMaxScaler. This scaling technique adjusts the score ranges to be within the interval [0,1] and does not alter the shape of the distributions.

Figure 1 shows the distribution of the original EchoNest features. We notice that the "liveness" and "speechiness" features initially have very skewed distributions. These distributions are improved after the log scaling transformation, as can be seen in Figure 2. The "danceability", "energy", and "valence" features already have either normal or even distributions, so the log scaling transformation does not significantly affect them.

3.3 Output tags, thresholding

Tags are generated based on calculated thresholds for EchoNest numeric features. For example, if the "Danceability" feature of an audio track is in the 80th percentile, the track is tagged as "Danceable". Similarly, if the "Valence" feature surpasses a specific threshold for an audio track, that track will be tagged as "Happy". The set of tags generated by these thresholds will be the labels for the multi-class classifier.

Threshold calculation: Thresholds are set to categorize the top percentage of tracks for each EchoNest feature. We use a threshold formula of $\mu + c \times \sigma$, where μ is the mean and σ is the standard deviation of the numeric EchoNest features. If a track has an EchoNest feature above this threshold, it will be assigned a 1 for the corresponding feature tag, and 0 otherwise. The threshold can be easily adjusted to optimize the tagging process, specifically by adjusting the coefficient c. In our experiments, we found that the value of $c = 0.5$ provided the best classification results. This essentially means that a track will be labeled with an EchoNest tag if its corresponding EchoNest score is at least half a standard deviation above the mean. The coefficient could have been included as an additional hyperparameter in the model evaluation, but in preliminary testing we found this to be unnecessary as all other values caused the classifier to produce much worse results.

3.4 Musical feature extraction

Input sound files are converted into a comprehensive set of features using the Librosa library. These features, which include Mel-Frequency Cepstral Coefficients (MFCC), Chroma, Tempo, and several others, form the inputs to the machine learning model. Below is a detailed table of the extracted features and their dimensionalities.

Table 1. Extracted Librosa Features

Features	Dimensions
Mel Spectrogram (Mean + std, n=64)	128
MFCC (Mean + std, n=40)	80
Chroma CQT (Mean + std, n=12)	24
Spectral Contrast (Mean + std, n=6)	12
Tonnetz (Mean + std)	12
RMS Energy (Mean + std)	2
Zero-Crossing Rate (Mean + std)	2
Tempo	1

The extracted features yield a dataset with X dimensions = (5200, 263) and Y dimensions = (5200, 7), where the number of input features for an audio file is 263, and the number of output classes is 7. Not all extracted features

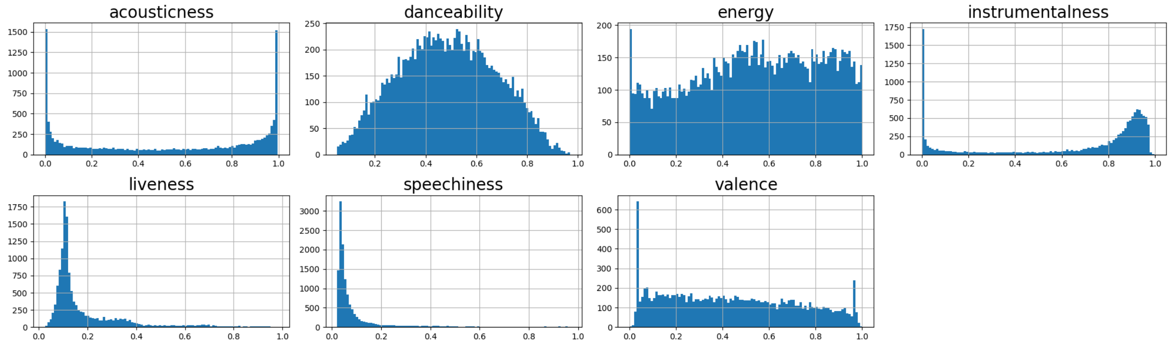

Figure 1. Distribution of original unit-range EchoNest scores.

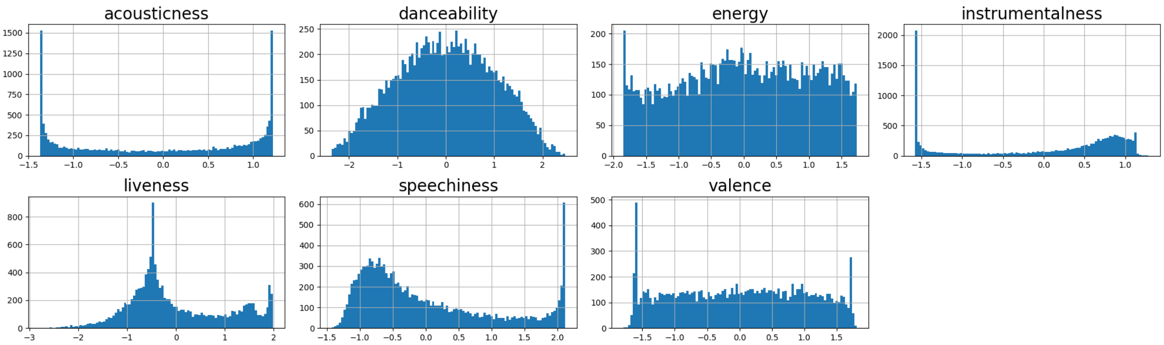

Figure 2. Distribution of Gaussian-normalized EchoNest scores.

are used directly in model training due to potential diminishing returns from including excessively many features, which can add noise and reduce model performance. To efficiently select the most informative features, we employ sklearn's SelectFromModel in conjunction with a RandomForest classifier. The RandomForest assesses feature importance, and SelectFromModel retains only those features that surpass a certain importance threshold. This threshold itself is treated as a hyperparameter in the model training stage, which we optimize using grid search to improve the classification metrics.

3.5 Training, data augmentation, tuning

To identify the most effective model for associating Librosa features with EchoNest tags, we expanded our testing by incorporating a larger dataset. Initially, our training dataset comprised approximately 800 tracks from the FMA_small dataset cross-referenced with the echonest.csv file. Later, we expanded on this by taking all tracks from FMA_medium that were included in the echonest.csv file. This expansion increased our training set to 5,300 tracks. Comparing the model trained on this larger dataset with the preliminary model showed minimal difference in results, suggesting that the quality of features extracted from audio files holds more significance than merely the quantity of training data. Future improvements will focus on deepening our understanding of each musical feature's relationship with output tags and refining the parameters used in Librosa's feature extraction functions.

For hyperparameter tuning, we used sklearn's GridSearchCV class to iterate through approximately 500 combinations of hyperparameters. This process is conducted using k-fold cross-validation only on the training set to ensure the model is tested on completely unseen data. The best hyperparameters identified for the RandomForest model include $max\_depth = 13$, $n\_estimators = 110$, $feature\_selection\_threshold = 0.7 * \mu$, while for the Multi-Layer Perceptron (Neural Network), they are $learning\_rate = 0.0005$, $hidden\_layer\_sizes = (10,)$, $feature\_selection\_threshold = 1.1 * \mu$. Notably, hyperparameter tuning did not significantly enhance the classification metrics, as can be seen in table 2, reinforcing the importance of feature selection and model training strategy over extensive parameter tuning.

3.6 Evaluation metrics

For model evaluation, we reserve a portion of the labeled dataset as a test set. We associate tag labels for each track in the test set using the predefined thresholds and compare these with the model's predictions. To assess the model's accuracy in predicting tags for unseen data, we use confusion matrices, accuracy, and F1-scores for each target tag. This approach provides a view of the model's performance across different tags, offering insights into the effectiveness of our chosen classifiers and feature sets.

4. EVALUATION

In assessing the performance of our classifier models, both the Random Forest and Multi-Layer Perceptron (MLP)

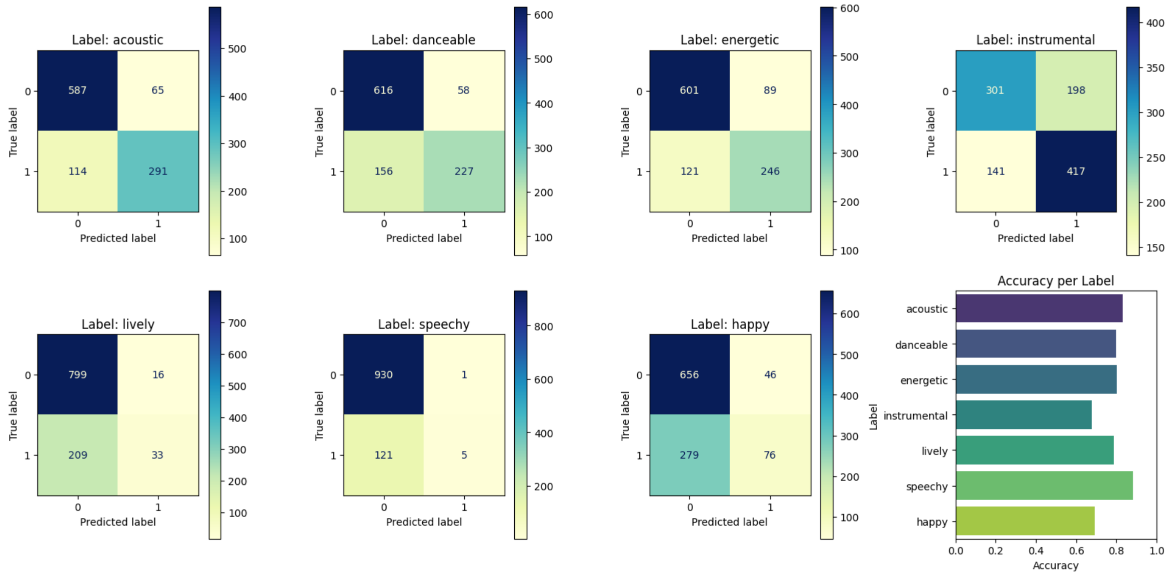

Figure 3. Final Random Forest model evaluation.

Table 2. Model Performance Comparison

Output Class	Random Forest		Multi-layer Perceptron		Stratified Dummy	
	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
Acoustic	0.83	0.76	0.84	0.78	0.52	0.40
Danceable	0.80	0.69	0.79	0.69	0.56	0.33
Energetic	0.80	0.69	0.81	0.72	0.56	0.33
Instrumental	0.68	0.64	0.66	0.62	0.49	0.52
Lively	0.79	0.24	0.78	0.32	0.59	0.24
Speechy	0.88	0.38	0.88	0.41	0.59	0.11
Happy	0.69	0.42	0.68	0.47	0.52	0.31
Average	0.78	0.55	0.78	0.57	0.55	0.32

showed notably similar results. Each model achieved an average accuracy per tag of 0.78 and an average F1 score around 0.56. Interestingly, our effort to expand the dataset from 800 to 5,300 tracks did not show substantial improvements in model performance. These outcomes highlight the importance of the quality of the extracted features over the quantity of data or extensive model tuning.

We believe further improvements to the auto-tagging model are likely to come from more meticulous feature extraction and engineering. Specifically, analyzing how different audio features are associated with EchoNest tags could unlock higher performance, particularly for tags such as 'Lively', 'Instrumental', and 'Happy'.

In comparison, the performance of a stratified dummy classifier, which predicts based on a random sample of the output class distribution, was considerably lower. This baseline model achieved an average accuracy per tag of 0.55 and an average F1 score of 0.36. This shows that our models are approximately 40% more effective than this baseline approach. Such a margin validates the effectiveness of a machine learning approach for associating simple musical features with EchoNest tags, and shows future potential for building an open-source model that fully replicates the original EchoNest algorithm by predicting scores.

5. RELATED WORKS

A paper by Choi, Keunwoo, et al. [5] introduces a convolutional recurrent neural network (CRNN) for music tagging. The algorithm separates songs into four general categories (Genre, Mood, Instrument, Era), and within them there are multiple subcategories. This work will serve as a great benchmark for comparison as it has a very similar outcome. Unlike our approach, the CRNN model integrates recurrent layers that capture temporal dependencies, while our model focuses on feature association through a multiclass classifier that does not explicitly model time sequences.

A paper by Turnbull, Douglas, et al. [2] discusses ways of classifying semantically meaningful words to different audio. This project will have a similar function but associate different music to a variety of tags. Our approach, however, diverges by focusing more on the direct extraction of numeric features and their correlation with predefined tags, rather than semantic classification of words.

A paper by Kim, Jong Wook, et al. [6] explains CREPE, a pitch estimation algorithm for monophonic audio recordings. It uses a pretrained convolutional neural network and outperforms existing algorithms for pitch estimation. We differ in that we do not concentrate on pitch but rather on a broader set of features, aiming to categorize music tracks into multiple descriptive tags.

A paper by Böck et al. [7] introduces madmom, a Python library for audio processing and music information retrieval (MIR), emphasizing its concise design, compatibility with NumPy, and object-oriented structure for efficient prototyping of MIR applications. While we may utilize this library's features, we focus on integrating these features into a unique classification model that combines multiple libraries and methodologies for enhanced tagging accuracy.

A paper by Salamon, Justin, and Emilia Gómez. [8] presents a system for the extraction of melody from polyphonic audio recordings. Their approach is based on the creation and characterization of pitch contours. This method out-performed existing approaches at the time of writing. In contrast, we emphasize not just melody but a diverse array of features to provide comprehensive tagging based on EchoNest features.

A paper by G. Liu et al. [9] talks about low accuracy in emotion classifiers for different audio. They then discuss how the fusion of lyrics and audio may be the solution to this problem. We could potentially implement something similar but our current approach focuses on purely audio-based features without incorporating lyrical content, aiming to refine the predictive power of musical features alone.

A journal article by J. Lee and J. Nam outlines how they used a convolutional neural network (CNN) for the task of music auto-tagging [10]. Their approach included three steps: supervised feature learning on audio features using CNNs of varying input sizes, use of features from each layer of the pretrained CNNs to form one long audio track, and the aggregation of this track into a fully connected CNN for predictions. Unlike their approach, we utilize a combination of feature selection techniques and classifiers to optimize tagging without relying on a single CNN architecture.

Another work by J.lee et al. explores using small portions of raw wave data as input to a deep CNN [11]. This method breaks from using a larger frame of data as input and uses sample-level filters. Our methodology contrasts by employing a variety of extracted features from Librosa rather than focusing on raw waveform data, aiming to find associations between musical characteristics and EchoNest features.

A research article by J. Liu and Y. Yang looks at applying auto-tagging to subsets of an audio clip, rather than the full length of a clip [12]. Their motivation is to allow for new ways for people to interact with music, similar to advances in computer vision that allow for the localization of visual objects. Our approach could potentially benefit from using their technique to apply our model to different segments of audio tracks and averaging the scores across segments to get more robust outputs.

A work by Y. Lin and H. Chen proposes a new methodology for improving the tags used to train auto-tagging models [13]. They focus on deriving context information about a piece of music and finding similar songs based on that context. Our approach similarly seeks to enhance the accuracy of tags but does so by refining feature extraction and classifier tuning rather than contextually redefining the tags themselves.

An article by G. Song et al. emphasizes the benefit of using a recurrent neural network to preserve information past the initial feature extraction pre-processing [14]. This method was motivated by the information loss that occurs when using feature extraction techniques such as MFCC. This contrasts with our methodology, which combines multiple feature extraction techniques and focuses on optimizing the selection process to minimize information loss. However, further improvements to our model may benefit from exploring the techniques introduced in this work to build stronger associations between audio tracks as a whole and EchoNest scores.

A paper by S. Wang et al. describes the creation of a dataset that has audio tracks with time annotated tags [15]. This kind of dataset is beneficial to the kind of research described in the previously mentioned article by J. Liu and Y. Yang in which they look at the task of training models to tag subsections of an audio track [12]. Our dataset, however, consists of EchoNest scores for entire audio tracks instead of segments. Therefore, we take a more holistic approach in associating tags to audio tracks.

6. REFERENCES

[1] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "FMA: A dataset for music analysis," in *18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017. [Online]. Available: https://arxiv.org/abs/1612.01840

[2] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet, "Semantic annotation and retrieval of music and sound effects," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 2, pp. 467–476, 2008.

[3] B. McFee, M. McVicar, D. Faronbi, I. Roman, M. Gover, S. Balke, S. Seyfarth, A. Malek, C. Raffel, V. Lostanlen, B. van Niekirk, D. Lee, F. Cwitkowitz, F. Zalkow, O. Nieto, D. Ellis, J. Mason, K. Lee, B. Steers, E. Halvachs, C. Thomé, F. Robert-Stöter, R. Bittner, Z. Wei, A. Weiss, E. Battenberg, K. Choi, R. Yamamoto, C. Carr, A. Metsai, S. Sullivan, P. Friesch, A. Krishnakumar, S. Hidaka, S. Kowalik, F. Keller, D. Mazur, A. Chabot-Leclerc, C. Hawthorne, C. Ramaprasad, M. Keum, J. Gomez, W. Monroe, V. A. Morozov, K. Eliasi, nullmightybofo, P. Biberstein, N. D. Sergin, R. Hennequin, R. Naktinis, beantowel, T. Kim, J. P. Åsen, J. Lim, A. Malins, D. Hereñú, S. van der Struijk, L. Nickel, J. Wu, Z. Wang, T. Gates, M. Vollrath, A. Sarroff, Xiao-Ming, A. Porter, S. Kranzler, Voodoohop, M. D. Gangi, H. Jinoz, C. Guerrero, A. Mazhar, toddrme2178, Z. Baratz, A. Kostin, X. Zhuang, C. T. Lo, P. Campr, E. Semeniuc, M. Biswal, S. Moura, P. Brossier, H. Lee, and W. Pimenta, "librosa/librosa:

0.10.1," Aug 2023. [Online]. Available: https://doi.org/10.5281/zenodo.8252662

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[5] 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.

[6] J. W. Kim, J. Salamon, P. Li, and J. P. Bello, "Crepe: A convolutional representation for pitch estimation," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 161–165.

[7] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, "Madmom: A new python audio and music signal processing library," in *Proceedings of the 24th ACM international conference on Multimedia*, 2016, pp. 1174–1178.

[8] J. Salamon and E. Gómez, "Melody extraction from polyphonic music signals using pitch contour characteristics," *IEEE transactions on audio, speech, and language processing*, vol. 20, no. 6, pp. 1759–1770, 2012.

[9] G. Liu and Z. Tan, "Research on multi-modal music emotion classification based on audio and lyirc," in *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, vol. 1. IEEE, 2020, pp. 2331–2335.

[10] 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.

[11] 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.

[12] J.-Y. Liu and Y.-H. Yang, "Event localization in music auto-tagging," in *Proceedings of the 24th ACM international conference on Multimedia*, 2016, pp. 1048–1057.

[13] Y.-H. Lin and H. H. Chen, "Tag propagation and cost-sensitive learning for music auto-tagging," *IEEE Transactions on Multimedia*, vol. 23, pp. 1605–1616, 2020.

[14] 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.

[15] S.-Y. Wang, J.-C. Wang, Y.-H. Yang, and H.-M. Wang, "Towards time-varying music auto-tagging based on cal500 expansion," in *2014 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2014, pp. 1–6.