Deep Learning Based EDM Subgenre Classification using Mel-Spectrogram and Tempogram Features

Wei-Han Hsu, Bo-Yu Chen, and Yi-Hsuan Yang

Research Center for IT Innovation

Academia Sinica

Taipei, Taiwan

{ddmanddman, bernie40916, yang}@citi.sinica.edu.tw

Abstract—Along with the evolution of music technology, a large number of styles, or "subgenres," of Electronic Dance Music (EDM) have emerged in recent years. While the classification task of distinguishing between EDM and non-EDM has been often studied in the context of music genre classification, little work has been done on the more challenging EDM subgenre classification. The state-of-art model is based on extremely randomized trees and could be improved by deep learning methods. In this paper, we extend the state-of-art music auto-tagging model "short-chunk CNN+Resnet" to EDM subgenre classification, with the addition of two mid-level tempo-related feature representations, called the Fourier tempogram and autocorrelation tempogram. And, we explore two fusion strategies, early fusion and late fusion, to aggregate the two types of tempograms. We evaluate the proposed models using a large dataset consisting of 75,000 songs for 30 different EDM subgenres, and show that the adoption of deep learning models and tempo features indeed leads to higher classification accuracy.

Index Terms—EDM subgenre classification, deep learning, feature fusion, tempogram, convolutional neural network

I. INTRODUCTION

Electronic Dance Music (EDM) is a kind of dance and club music. Disk Jocket (DJs) usually need to classify EDM by their subgenres to get songs with similar style for many purposes, such as for making the DJ set transitions part. An automatic program for EDM subgenre classification can be useful for human DJs [1]. It is also a fundamental building block towards realizing an automatic AI DJ [2], [3].

The task *automatic EDM subgenre classification* can be in general considered as an instance of music auto-tagging problem [4]. Therefore, methodology-wise we can base on research that has been done for general music auto-tagging and classification. However, we note that, due to the similarity among the EDM subgenres, EDM subgenre classification can sometimes be difficult even for human DJs. The decision between genres can be fuzzy. For example, "tech-house," "deep-house," and "progressive-house" music may sound fairly similar as they are all "house" music.

We present in this paper a deep learning based approach to automatic EDM subgenre classification, extending the recent work by Caparrini *et al.* [5], which use a non-deep learning approach. Following their work, we compile our training and test sets from Beatport (https://www.beatport.com/), a worldwide principal source of music for DJs. The Beatport website assigns only a single subgenre label to each song,
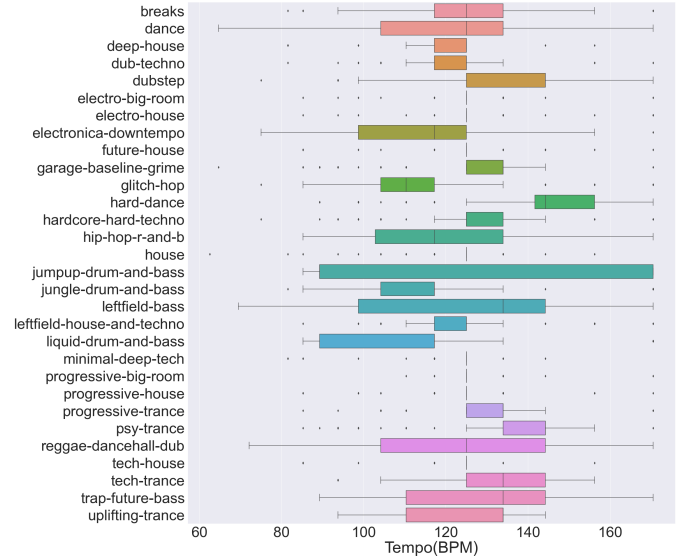

Fig. 1. Boxplots of the tempo values (in beat-per-minute; or BPM) of different EDM subgenres considered in our work.

so we can formulate the task as a multi-class classification problem. Our dataset contains 30 different subgenres, each with 2,500 songs and hence 75,000 songs in total. We thus treat it as a 30-class classification problem.

While the classifier used by Caparrini *et al.* [5] was based on extremely randomized trees [6], a non-deep learning algorithm, our classifier is based on the "short-chunk convolutional neural network (CNN)+Resnet" deep architecture proposed by Won *et al.* [7], which represents the state-of-the-art in music auto-tagging (see Figure 2). However, the original short-chunk CNN model takes only the Mel-spectrograms as input for feature learning. While this may be sufficient for classifying broader genre classes such as Pop, Rock, Jazz, and EDM, it is unclear how it performs for a subgenre classification task.

In particular, we note that, among the 92 hand-crafted audio features employed by Caparrini *et al.* [5], *tempo*-related features were found to be the top-four important features. By estimating the tempo values for all the songs in our dataset using the algorithm of Grosche *et al.* [8], and plotting the resulting distribution of tempo values for 100 randomly picked songs from each genre, as shown in Figure 1, we can see that

different EDM subgenres do prefer different tempo values. For example, the tempo of "psy-trance" music is usually around 140 beat-per-minute (BPM), while "deep-house" is usually around 120 BPM. Different subgenres also have different tempo ranges. For example, the tempo of "jumpup-drum-and-bass" can vary widely from 90 to 170 BPM, while the tempo values of "tech-house" typically fall within 120 to 130 BPM.

In light of this, we propose simple extensions of the short-chunk CNN+Resnet model [7] to learn features from not only the Mel-spectrogram but also the "tempogram" [8], a feature representation of the possible tempo values of each short-time frame of a signal. We consider both the autocorrelation- and Fourier-tempogram proposed by Grosche et al. [8], and experiment with both an early-fusion architecture and a late-fusion architecture. Our experiment shows that the Mel-spectrogram only baseline attains 55.4% song-level accuracy, and that our best model incorporating additionally tempograms achieves 60.6% accuracy. We find in particular salient improvement in subgenres such as "future-house," "leftfield-house-and-techno," and "uplifing-trance".

While we cannot re-distribute the audio files of the data due to copyright issues, we release our code and model checkpoints at https://github.com/mir-aidj/EDM-subgenre-classifier.

II. RELATED WORK

Music genre classification is a well-researched task in the field of music information retrieval (MIR), usually formulated as a multi-class classification problem. For example, the most famous and widely-used dataset in music genre classification, the GTZAN dataset [9], includes in total 1K songs covering the following 10 genres: Blues, Classical, Country, Disco, Hiphop, Jazz, Metal, Pop, Reggae and Rock. In contrast, relatively less research has been done on music subgenre classification, which aims at distinguishing between subgenres belonging to the same "super" genre such as EDM and Jazz. We review three such existing work below.

EDM subgenre classification. Caparrini et al. [5] presented a non-deep learning method for EDM subgenre classification. They compiled two different collections of EDM subgenre data from Beatport, the first with 23 subgenres and the second expanded version with 29 subgenres. They retrieved 100 songs for each subgenre from the Beatport website for both datasets. Then, they employed 92 hand-crafted audio features and tested several non-deep learning based classifiers under 10-fold cross validation. For the first set, the gradient tree boosting [10] performed the best, reaching 59.2% song-level classification accuracy. For the second set, which is more challenging due to the larger number of classes, the extremely randomised trees [6] performed the best, reaching 48.2% accuracy. They also drew two confusion matrices for both two sets, as well as the directed graph to show which subgenres were easily misclassified. And, they measured the importance of each feature in their tree-based classifier, finding that the top-four important features are all related to tempo.

Heavy metal subgenre classification. Tsatsishvili [11] investigated subgenre classification for heavy metal music,
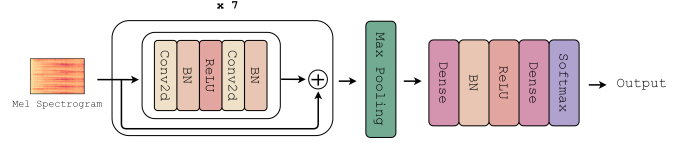

Fig. 2. System diagram of the short-chunk CNN+Resnet model [7].

also using non-deep learning methods. The author compiled a dataset of 210 songs, comprising 30 songs for each of the following 7 heavy metal subgenres: "black," "death," "melodic," "death," "gothic," "heavy," "power" and "progressive." Half the songs were used for training and the other half for testing. Audio features employed in the classifiers were automatically chosen from 200 hand-crafted features with either a correlation-based feature selection method or a wrapper selection method. The best classification accuracy was achieved by AdaBoost [12], reaching 45.7%.

Jazz subgenre classification. Quinto et al. [13] employed deep learning methods for Jazz subgenre classifcation, considering only three subgenres: "acid-jazz," "bebop," and "swing/electroswing." They considered a simpler multi-layer perceptron (MLP) with 1–3 layers, as well as a more sophisticated 3-layer recurrent neural network employing long short-term memory (LSTM) with 32 neurons per layer. They employed Mel-frequency cepstral coefficients (MFCC) as the input feature. Their dataset includes 254 minutes of "acid-jazz," 141 minutes "bebop," and 245 minutes "swing/electroswing." 60% of them were used for training and the rest for testing. Finally, they got 90% testing accuracy with the LSTM classifier, and 79% with the MLP classifier. The accuracy was fairly high, possibly because of the small number of classes.

Unlike music genre/subgenre classification, the **music auto-tagging** task [7], [14]–[20] is often formulated as a multi-label classification problem, where a song can be labeled with multiple tags. The performance of an auto-tagging model is usually evaluated in terms of metrics such as PR-AUC and ROC-AUC [4]. Won et al. [7] experimented with a larger number of different CNN-based models on three widely-used public datasets for this task: MagnaTagATune (MTAT) [14], million song dataset (MSD) [21], and MTG-Jamendo [19], finding that a particular architecture called "short-chunk CNN+Resnet" performs in general the best. As it is straightforward to modify this architecture for multi-class classification, we employ it as the backbone architecture of our models in this work.

III. METHODS

A. Dataset

Following Caparrini et al. [5], we build our dataset by crawling the audio previews of songs and their corresponding class labels from Beatport. Different from their work, we consider a larger set of classes, covering the 30 EDM subgenres listed in Figure 1, and a much larger collection of songs, with consistently 2,500 songs per subgenre.[1] We split the dataset by

---

[1]This Beatport dataset was initially created in one of our prior works [24].
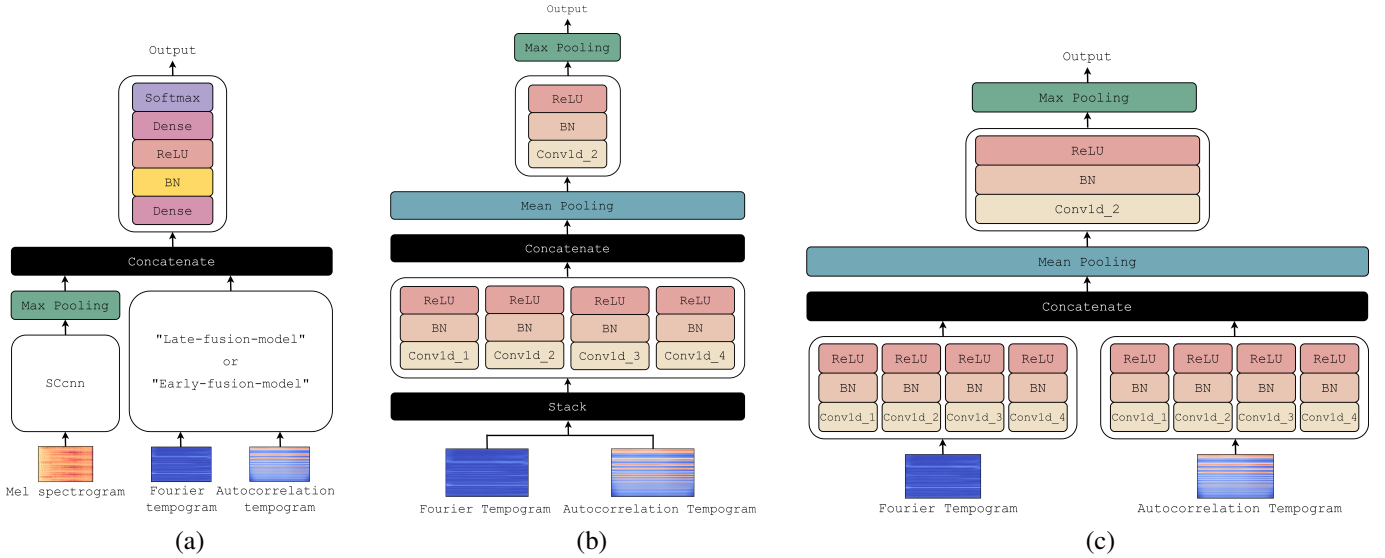
Fig. 3. System diagram of the proposed architecture for (a) employing both Mel-spectrograms and two different tempograms for music subgenre classification; (b) zoom-in of the early-fusion module for fusing the two tempograms; (c) the late-fusion variant. The 'SCcnn' in (a) denotes the stack of 7 copies of feature extraction blocks (each with two 2-D convolutions) shown in Figure 2.

the ratio of 8:1:1 per subgenre to get the training, validation, and test sets. Namely, the test set contains 250 tracks per subgenre. Every audio preview made available by Beatport is 2 minutes long. We make them consistently mono channel with 22,050 Hz sampling rate.

B. Input Features: Mel-spectrogram and Tempograms

Instead of using hand-crafted features as done by Caparrini *et al.* [5], we employ "raw" features as input to our deep neural network for feature learning. The basic feature representation, also one of the most widely-used one in MIR, is the Mel-spectrogram, a time-frequency representation that is computed by applying the perceptually-motivated Mel filter bank to the spectrogram of an audio waveform. With librosa [22], we compute the Mel-spectrogram using a Hamming window of 2,048-sample long and 512-sample hop length for the short-time Fourier transform (STFT), and 128 Mel filters.

We also employ the tempogram [8] as input feature, a "time-tempo" representation that contains local tempo information for each frame of an audio signal. Grosche *et al.* [8] proposed two types of tempogram: the *Fourier tempogram* and the *autocorrelation tempogram*. The former converts frequency (Hz) to tempo (beat-per-minute; BPM), emphasizing the harmonics, while the latter converts time-lag (seconds) to tempo, emphasizing instead the subharmonics. We consider both here. The *Fourier tempogram*[2] is computed by firstly estimating from the audio waveform a "novelty curve" indicating note onset candidates, and then computing the Fourier representation of the novelty curve (not the original audio waveform) using STFT. The resulting representation is assumed to capture local periodic patterns of the input signal. Its frequency

axis is finally mapped to a BPM axis. The *autocorrelation tempogram*,[3] on the other hand, is computed by calculating the local autocorrelation function for different time lags from the also the novelty curve, and then converting the time-lag to a linear BPM axis using interpolation and resampling. We also employ librosa [22] to compute these two variants of the tempogram, with 512-sample hop size and a Hamming window of 2,048 samples. Figure 6 provides examples of the Fourier tempogram and autocorrelation tempogram of songs in our dataset.[4]

We use two different length per song to compute the Mel-spectrograms: the 30-second segment from 15s to 45s of a song, or the whole two minutes. For the tempograms, we use only the segment a song from 15s to 45s for simplicity. The size of a Fourier tempogram and an autocorrelation tempogram would be $193\times1{,}293$ (BPM$\times$time) and $384\times1{,}292$, respectively. Audio is fed to our models using "chunks" (see below) of such fixed-length Mel-spectrograms and tempograms. Every feature dimension is zscore-normalized.

C. Short-chunk CNN with Resnet

We employ the short-chunk CNN+Resnet as the backbone architecture of our models, as it has been shown to outperform competing models in music auto-tagging across different benchmark datasets [7]. As depicted in Figure 2, this model contains 7 copies of feature extraction layers, each comprising two 2-D convolutional layers, two batch normalization layers (after convolution), and one ReLU activate layer in between convolutions. The output of this stack of layers goes through max pooling and then two dense layers and a final softmax

---

[2]https://www.audiolabs-erlangen.de/resources/MIR/FMP/C6/C6S2_TempogramFourier.html

[3]https://www.audiolabs-erlangen.de/resources/MIR/FMP/C6/C6S2_TempogramAutocorrelation.html

[4]Grosche *et al* [8] also proposed the "cyclic tempogram," where tempi differing by a power of two are identified, but we do not use it here.

TABLE I
CHUNK- AND SONG-LEVEL TESTING ACCURACY OF 30-CLASS EDM
SUBGENRE CLASSIFICATION OF THE EVALUATED SHORT-CHUNK
CNN-BASED MODELS; THE FIRST TWO MODELS USE ONLY THE
MEL-SPECTROGRAMS, WHILE THE LAST TWO USE BOTH THE
MEL-SPECTROGRAMS AND TEMPOGRAMS

	chunk-level	song-level
Mel-spectrogram only (30 sec) [7]	46.1%	50.4%
Mel-spectrogram only (120 sec) [7]	46.1%	55.4%
Fourier tempogram only (30 sec)	32.0%	34.9%
autocorrelation tempogram only (30 sec)	28.3%	31.2%
early-fusion	53.4%	60.3%
late-fusion	53.3%	60.6%

layer for classification. The name "short-chunk" stems from the fact that the model is designed to take as input short segments of the Mel-spectrogram of the size 128×200, which in other words divides our Mel-spectrogram into 25 chunks (neglecting the last 168 frames). We similarly divide the tempograms into 25 chunks. Following [7], we assume that the subgenre label of each chunk is the same as the the subgenre label of the song the chunk comes from.

D. Proposed Fusion Models

Figure 3(a) shows the architecture we propose to integrate the (chunk-level) Mel-spectrogram, Fourier tempogram, and autocorrelation tempogram of a song for classification. We firstly fuse the two tempograms into a combined representation, and then concatenate it with the output of the feature extraction blocks of the short-chunk CNN branch that deals with the Mel-spectrogram. After feature concatenation, we use the same classification block of the short-chunk CNN.

Figures 3(b) and (c) depict the two fusion strategies considered in our work to combine the two tempograms. They both use four parallel 1-D convolutional layers with different kernel sizes (3, 3, 5, 5) and strides (2, 3, 3, 5) for feature extraction, a design that is inspired by the work of Pons *et al.* [23]. The outputs of the 1-D convolution layers are concatenated, mean-pooled, futher processed with a 2-D convolution layer, and finally max-pooled to yield a combined representation. The *early-fusion* variant (Figure 3(b)) combines the two tempograms at the very beginning with feature-wise concatenation, while the *late-fusion* variant (Figure 3(c)) combines them after the 1-D convolutional layers.

IV. EXPERIMENTS

We train the following deep models by using cross-entropy as the loss function and Adam as the optimizer.

- *Mel-spectrogram only*: the short-chunk CNN+Resnet baseline [7] shown in Figure 2.
- *Proposed early- and late-fusion*: the proposed models that use both the Mel-spectrograms and the two tempograms, shown in Figure 3.
- *Fourier tempogram or autocorrelation tempogram only*: the ablated variants that use only one of the tempograms, without using the Mel-spectrograms. This is done with
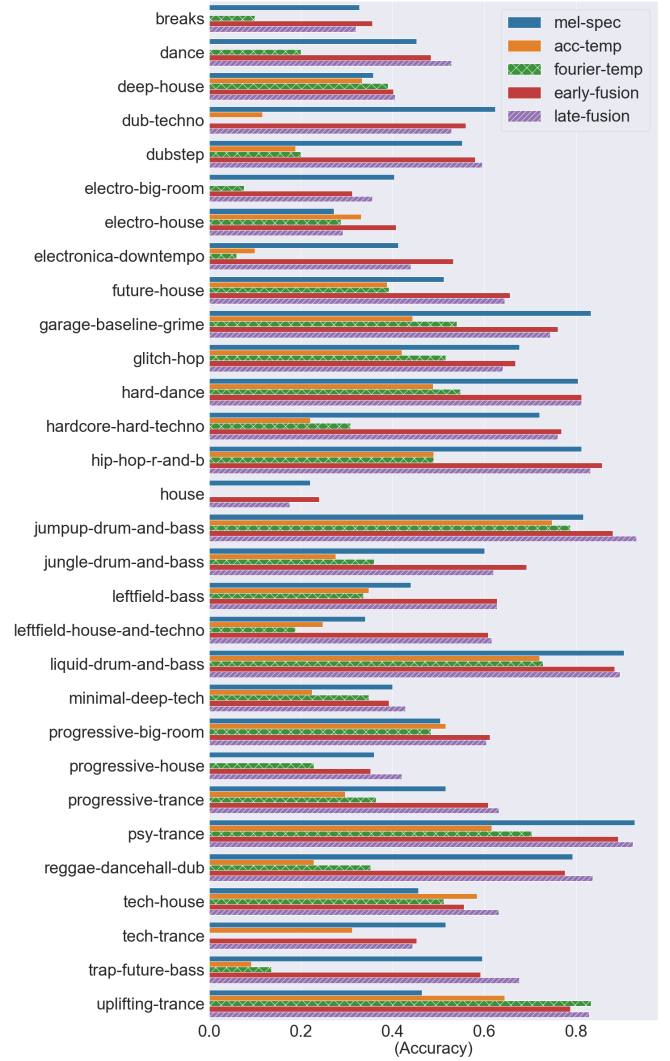

Fig. 4. Per subgenre testing classification accuracy of different models.

a simplified version of the fusion models, with four 1-D convolutional layers and a 2-D convolutional layer for feature extraction, and two dense layers for classification.

We set the batch size to 256, and train the models for 200 epochs with learning rate 0.005. We use 50% dropout before the last dense layer and after the ReLU layer.

We report the **chunk-level** classification accuracy by taking the class with the highest softmax-ed value as the prediction result. In addition, we also report the **song-level** accuracy by a majority voting mechanism over the prediction results of the short chunks of a song.

Table I shows that early-fusion and late-fusion models are both better than Mel-spectrogram-input model in both chunk-level and song-level accuracy. The late-fusion model performs overall the best, reaching 60.6% song-level testing accuracy for our 30-class classification setting, which is much higher than the 48.2% accuracy obtained by Caparrini *et al.* [5] for 29-class classification using a non-deep learning method.This also stands as a +5 to +10% performance gain relative to the Mel-

(a) Confusion table of the short-chunk CNN baseline (120 sec)

(b) Confusion table of the proposed late-fusion model

Fig. 5. The confusion matrices of the testing result of two evaluated models. Each row shows whether the songs from a subgenre are misclassified to other subgenres; those on the diagonal are correctly classified. As our test set contains consistently 250 songs per subgenre, each row sums up to 250.

spectrogram only baseline, empirically validating the benefit of using tempo-related features for EDM subgenre classification. Table I also shows that the Fourier tempogram only model outperforms the autocorrelation tempogram counterpart.

Figure 4 shows the per-genre result of these five models. We can see salient performance improvement for genres such

as "future-house," "leftfield-house-and-techno" and "uplifing-trance" for the proposed fusion models than the baseline Mel-spectrogram only model.

Figure 5 shows the 30×30 confusion matrices of the baseline model and the proposed late-fusion model. We can see that the propose model still gets confused for some pairs of subgenres, but less seriously so than the baseline model.

Finally, Figure 6 visualizes the Mel-spectrograms and tempograms of four songs from our test set. The songs shown in Figures 6(a) and (b) are both "uplifting-trance" songs. They can both be correctly classified by our late-fusion model, but the first song would be wrongly recognized as a "tech-trance" song by the baseline model. We see that these two songs seem to have fairly different patterns in the Mel-spectrograms, but similar patterns in the autocorrelation tempograms. The baseline model does not work well for this song, possibly because it only has access to the Mel-spectrograms. On the other hand, the songs shown in Figures 6(c) and (d) are an "uplifing-trance" song and a "tech-trance" song, respectively. Again, they can both be correctly classified by our late-fusion model, but the first song in this pair (i.e. (c)) would be wrongly regarded as a "tech-trance" song by the baseline model. We see these two songs, despite they are associated with different subgenres, have some similar local patterns in their Mel-spectrograms, which might have caused confusion for the baseline model.

V. CONCLUSION

In the paper, we have presented a deep learning approach for EDM subgenre classification, achieving 60.6% testing accuracy for 30-class classification when using not only the Mel-spectrograms but also the tempograms as input feature. We found that the proposed late-fusion model is 10% more accurate than a Mel-spectrogram only deep learning baseline model. This may suggest that, for a fine-grained classification problem such as EDM subgenre classification, it is beneficial to consider multiple inputs for feature learning.

We have a few ideas for future extension of this work. First, we would like to explore different architectures to process the tempograms, such as the use of recurrent layers or temporal convolutional layers [25]. Second, to incorporate more features as input, including other mid-level features [26] as well as high-level features [17], [24], [27]. Third, to employ other feature fusing strategies. And finally, to employ metric-based learning method [28], which is shown to be promising in recent work on music auto-tagging.

REFERENCES

[1] S. Sankalp, T. Baruah, S. Tiwari and S. Ganesh, "Intelligent classification of electronic music," in *Proc. IEEE Int. Symposium on Signal Processing and Information Technology*, pp. 31–35, 2014.

[2] Y.-S. Huang, S.-Y. Chou, and Y.-H. Yang, "DJnet: A dream for making an automatic DJ," in *Proc. Int. Soc. Music Information Retrieval Conf.*, late-breaking and demo paper, 2017.

[3] B.-Y. Chen, W.-H. Hsu, W.-H. Liao, M. A. M. Ramírez, Y. Mitsufuji, and Y.-H. Yang, "Automatic DJ transitions with differentiable audio effects and generative adversarial networks," in *arXiv preprint:2110.06525*, 2021.
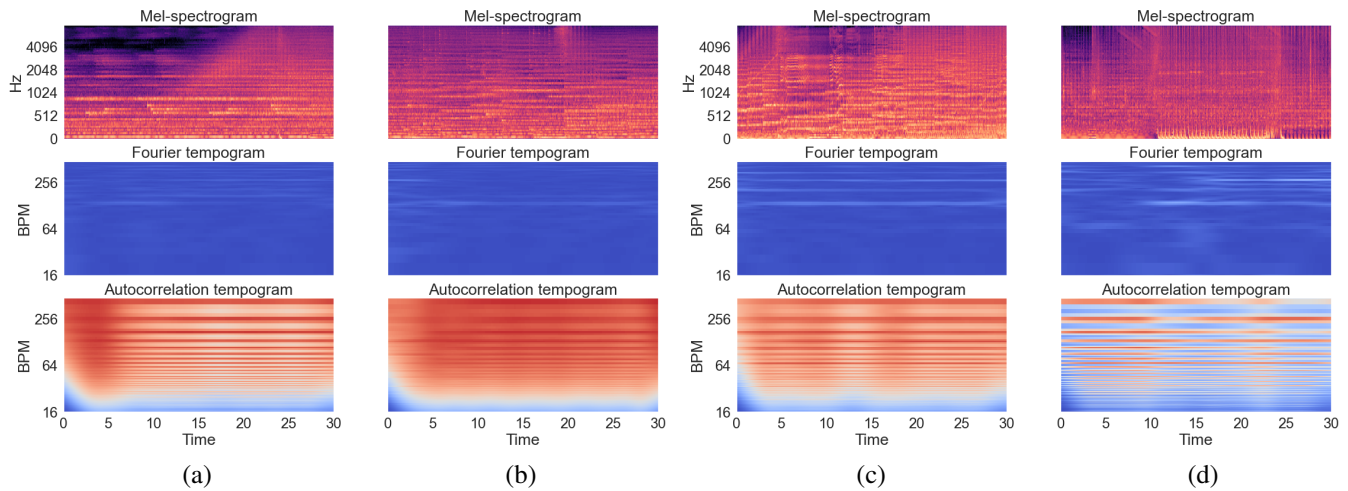
Fig. 6. The Mel-spectrogram, Fourier tempogram and autocorrelation tempogram for (a) an "uplifting-trance" song misclassified as "tech-trance" by the baseline model but correctly classified by our late-fusion model; (b) an "uplifting-trance" song correctly classified by both the baseline model and our fusion model; (c) another "uplifting-trance" song misclassified as "tech-trance" by the baseline model but correctly classified by our late-fusion model; (d) a "tech-trance" correctly classified by both the baseline model and our fusion model.

[4] J. Nam, K. Choi, J. Lee, S.-Y. Chou, and Y.-H. Yang, "Deep learning for audio-based music classification and tagging," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 41–51, Jan. 2019.

[5] A. Caparrini, J. Arroyo, L. Pérez-Molina, J. Sánchez-Hernández, "Automatic subgenre classification in an electronic dance music taxonomy," *Journal of New Music Research*, vol. 49, no. 3, pp. 269–284, 2020.

[6] P. Geurts, D, Ernst, L. Wehenkel, "Extremely randomized trees," *Proc. Machine learning*, pp. 3–42, 2006.

[7] M. Won, A. Ferraro, D. Bogdanov, X. Serra, "Evaluation of CNN-based automatic music tagging models," in *Proc. Sound and Music Computing Conf.*, 2020.

[8] P. Grosche, M. Müller, and F. Kurth, "Cyclic tempogram: A midlevel tempo representation for music signals," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 5522–5525, 2010.

[9] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, Jul. 2002. [Online] https://www.kaggle.com/andradaolteanu/gtzan-dataset-music-genre-classification.

[10] J. H. Friedman, "Stochastic gradient boosting," *Proc. Computational statistics & data analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[11] V. Tsatsishvili, "Automatic subgenre classification of heavy metal music," University Of Jyväskylä, master thesis, 2011.

[12] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, B. Kégl, "Aggregate features and adaboost for music classification," *Proc. Machine learning*, vol. 65, no. 2-3, pp. 473–484, 2006.

[13] R. J. M. Quinto, R. O. Atienza, and N. M. C. Tiglao, "Jazz music subgenre classification using deep learning," in *Proc. IEEE Region 10 Conf.*, pp. 3111–3116, 2017.

[14] E. Law, K. West, M. I. Mandel, M. Bay, and J. S. Downie, "Evaluation of algorithms using games: The case of music tagging." in *Proc. Int. Soc. Music Information Retrieval Conf.*, pp. 387-–392, 2009.

[15] 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, Feb. 2008.

[16] J.-Y. Liu and Y.-H. Yang, "Event localization in music auto-tagging," in *Proc. ACM Multimedia*, pp. 1048–1057, 2016.

[17] J. Pons and X. Serra, "musicnn: Pre-trained convolutional neural networks for music audio tagging," in *arXiv preprint:1803.01271*, 2018.

[18] M. Won, S. Chun, O. Nieto Caballero, X. Serra, "Automatic music tagging with harmonic CNN," in *Proc. Int. Soc. Music Information Retrieval Conf.*, 2019.

[19] D. Bogdanov, M. Won, P. Tovstogan, A. Porter, X. Serra, "The MTG-Jamendo dataset for automatic music tagging," in *Proc. Machine Learning for Music Discovery Workshop*, 2019.

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

[21] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proc. Int. Soc. Music Information Retrieval Conf.*, 2011.

[22] B. McFee, C. Raffel, D Liang, D. P. Ellis, M. McVicar, E. Battenberg, O. Nieto, "librosa: Audio and music signal analysis in Python," in *Proc. Python in Science Conf.*, vol. 8, pp.18–25, 2015.

[23] J. Pons, O. Slizovskaia, and R. Gong, "Timbre analysis of music audio signals with convolutional neural networks," in *Proc. European Signal Processing Conf.*, pp. 2744–2748, 2017.

[24] Y.-S. Huang, S.-Y. Chou and Y.-H. Yang, "Pop music highlighter: Marking the emotion keypoints," *Transactions on International Society for Music Information Retrieval*, vol. 1, no. 1, pp. 68–78, 2018.

[25] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," in *arXiv preprint:1909.06654*, 2019.

[26] H. Foroughmand and G. Peeters, "Deep-Rhythm for global tempo estimation in music," in *Proc. Int. Soc. Music Information Retrieval Conf.*, pp. 636–643, 2019.

[27] E. Zangerle, M. Vötter, R. Huber, and Y.-H. Yang, "Hit song prediction: Leveraging low- and high-level audio features," in *Proc. Int. Soc. Music Information Retrieval Conf.*, pp. 319–326, 2019.

[28] M. Won, S. Oramas, O. Nieto, F. Gouyon, and X. Serra, "Multimodal metric learning for tag-based music retrieval," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pp. 591–595, 2021.