An Audio-Based Deep Learning Framework For BBC Television Programme Classification

Lam Pham[1,2], Chris Baume[3], Qiuqiang Kong[4], Tassadaq Hussain[2], Wenwu Wang[2], Mark Plumbley[2]

1. Center for Digital Safety & Security, Austrian Institute of Technology, Austria.
2. Center for Vision Speech and Signal Processing, University of Surrey, UK
3. BBC Research and Development, BBC, UK
4. ByteDance AI Lab, ByteDance, China

***Abstract*—This paper proposes a deep learning framework for classification of BBC television programmes using audio. The audio is firstly transformed into spectrograms, which are fed into a pre-trained Convolutional Neural Network (CNN), obtaining predicted probabilities of sound events occurring in the audio recording. Statistics for the predicted probabilities and detected sound events are then calculated to extract discriminative features representing the television programmes. Finally, the embedded features extracted are fed into a classifier for classifying the programmes into different genres. Our experiments are conducted over a dataset of 6,160 programmes belonging to nine genres labelled by the BBC. We achieve an average classification accuracy of 93.7% over 14-fold cross validation. This demonstrates the efficacy of the proposed framework for the task of audio-based classification of television programmes.**

***Index Terms*—Spectrogram, Convolutional Neural Network, Multilayer Perceptron, Support Vector Machine, Linear Regression, Decision Tree, Random Forest.**

I. Introduction

As the most popular media source in the UK, the BBC is used by more than 90% of adults every week [1]. We wish to develop an effective recommendation system that helps audiences find suitable programmes based on their interests and needs. Achieving an effective recommendation system not only requires a diverse user profile, but detailed metadata about the content. However, this is challenging for broadcasters as their content is neither segmented nor well-defined. Creating metadata manually is expensive, so an automatic tool to extract relevant metadata describing the content of BBC programmes reduces the cost. Given a BBC programme, diverse resources such as topic, video (image data), transcript (text data), or audio (acoustic data), which contain rich information, can be used for extracting metadata. We focus on analysing information extracted from audio, to evaluate whether both natural sounds and human speech detected in BBC programmes are useful for generating programme metadata. To evaluate the value of audio, we firstly propose a task of audio-based classification of television programmes in this paper which makes use of deep learning techniques. In particular, the task proposed is to classify audio recordings of BBC programmes into the nine BBC genre categories [2]: *Children's, Drama, Factual, Music, Sport, Weather, Comedy, Entertainment* and *News*. Given the deep learning classification model achieved in this paper, audio feature will be extracted and then integrated into the BBC metadata.

II. Background

Regarding recently proposed systems for broadcast media classification [3]–[8], authors made use of multiple features extracted from various resources such as audio, video, or transcript to maximize the system performance. For example, systems proposed in [6], [7] explored the programme context via semantic concepts of sunset, indoor, outdoor, cityscape, landscape, mountains, etc., and their relation among classified genres. Meanwhile, both audio and video features were made use in [3]–[5], [8]. To further improve a real-time system performance, an Automatic Speech Recognition (ASR) model was used in [9] to detect key words that enriches the word vectors representing a programme. Recently, Mortaza *et al.* [10] provided a comprehensive analysis of main features such as audio, text and the other metadata (channel, time) that were used for BBC broadcast classification. From the existing literature of audio feature extraction in broadcast classification [8], [10]–[12], we can see that these methods followed two main steps: Acoustic modeling on short-time segments and statistical modeling across the programme. For example, Ekenel *et al.* [8] firstly extracted Mel-frequency cepstral coefficients, fundamental frequency, signal energy, zero crossing rate from short-time segments split from each programme. Then, they applied a Gaussian Mixture Model (GMM) to learn these features, resulting in an embedded vector containing mean, standard deviation and weight of the GMM models. Similarly, in [10]–[12], short-time segments split from each audio recording were firstly transformed into vectors by using Linde–Buzo–Gray vector quantization algorithm [13]. These quantized vectors were then trained by a GMM model, resulting in embedded vectors containing GMM model information. Finally, latent Dirichlet allocation (LDA) based statistics, which have been used mostly in natural language processing (NLP) for the categorisation of text documents, were applied to map embedded vectors to the LDA domain. These audio feature extraction methods show how GMM was widely used for acoustic modeling

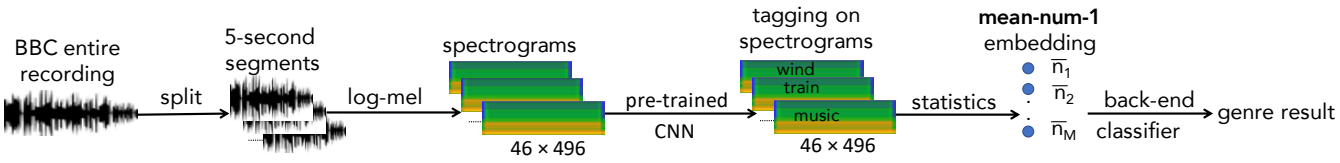

Fig. 1. Baseline architecture

TABLE I
THE PRE-TRAINED CNN BASED NETWORK ARCHITECTURE (INPUT IMAGE PATCH OF 64×496)

Network architecture	Output
Convolution [3×3@64] - BN - ReLU	64×496×64
Convolution [3×3@64] - BN - ReLU - Dropout (20%)	64×496×64
Convolution [3×3@128] - BN - ReLU	64×496×128
Convolution [3×3@128] - BN - ReLU - Dropout (20%)	64×496×128
Convolution [3×3@256] - BN - ReLU	64×496×256
Convolution [3×3@256] - BN - ReLU - Dropout (20%)	64×496×256
Convolution [3×3@512] - BN - ReLU	64×496×512
Convolution [3×3@512] - BN - ReLU	64×496×512
Average Pooling [2 × 2] - Dropout (30%)	32×248×512
Convolution [3×3@1024] - BN - ReLU	32×248×1024
Convolution [3×3@1024] - BN - ReLU - Dropout (30%)	32×248×1024
Convolution [3×3@2048] - BN - ReLU	32×248×2048
Convolution [3×3@2048] - BN - ReLU	32×248×2048
Global Pooling - Dropout (50%)	2048
FC - ReLU - Dropout (50%)	2048
FC - Sigmoid	$M = 527$

over short-time segments. However, short-time segments in each television programme may contain very different or very similar sound events. For example, a short-time segment may contain different natural sound events such as wind, bird song, or water fall sounds in *Factual* programmes, or only content human speech in *News* or *Weather* programmes. Therefore, using GMM for acoustic modeling over short-time segments may be ineffective with BBC programmes. Indeed, a performance comparison presented in [10] indicates that audio feature obtained low performance compared to the other features. Additionally, some television programmes have a long duration – often more than four hours for events such as live sports. This leads to a high cost for training GMM models. To deal with the limitations of GMM for acoustic modelling within the proposed task, a deep learning based framework is proposed. In particular, we firstly train a CNN based network with AudioSet dataset [14] defining M types of natural sounds and human speeches in life. Short-time segments split from each programme recording are transformed into spectrograms, then fed into the pre-trained CNN network to obtain the predicted probabilities for each of the M sound events defined in AudioSet dataset. We then conduct statistics over the predicted probabilities and sound events detected on all segments to generate embedded features representing each television programme. Given by the embedding extracted, we classify it into one of the nine different genres mentioned earlier.

III. BASELINE ARCHITECTURE

To evaluate the framework proposed, we firstly design a baseline system, outlined in Fig. 1. As shown in Fig. 1, the baseline proposed comprises four main steps: the spectrogram transformation, the pre-trained CNN–based network, statistics for embedded feature extraction, and the back-end classifier. Each is described below.

A. Log-mel Spectrogram Transformation

As the duration of BBC programmes can be very long, the programme recording is firstly split into non-overlapping 5-second segments which are suitable for back-end classifiers. The 5-second segments are then transformed into log-mel spectrograms of 46×496 by using Librosa toolbox [15]. We re-used settings from our previous work [16]: fMin = 50 Hz, fMax = 14000 Hz, sample rate = 32000 Hz, window size = 1024, hop size = 320, and Mel filter number = 64.

B. Pre-trained CNN–based Network

As a part of our previous work [16], the pre-trained CNN based network, which was trained with AudioSet dataset [14], is based on the VGG architecture [17] as shown in Table I. In particular, the architecture contains sub-blocks which perform convolution, batch normalization (BN) [18], rectified linear units (ReLU) [19], average pooling, global pooling [1], dropout [20], fully-connected (FC) and Sigmoid layers. The dimension of Sigmoid layer is set to $M = 527$ that equals to the number of sound events defined in AudioSet dataset [14]. In total, we have 12 convolutional layers and two fully-connected layers containing trainable parameters that makes the proposed CNN network like VGG-14 [17].

C. Statistics For Embedded Feature Extraction

Given the pre-trained CNN network architecture, when a 5-second log-mel spectrogram is fed into the network, the output of the Sigmoid layer, likely a M-dimensional vector, is obtained. Each dimension of the vector presents the predicted probability of one type of M sound events that may occur in each 5-second segment. In the baseline system proposed, we only select the sound event which shows the highest probability for extracting the embedded feature. In other words, each segment in a programme is now tagged by only one sound event with the highest probability, referred to as single-sound-event tagging information. Let us consider $\mathbf{n} = (n_1, n_2, \ldots, n_M)$ as the *number* vector, where M is the number of sound events defined in AudioSet dataset [14], and n_i is the total number of times that the ith sound event is detected and used to tag on segments in each programme.

[1]the global pooling used is a combination of both max and average global pooling [16]

TABLE II
DIFFERENT BACK-END CLASSIFIER EVALUATED.

Classification Models	Setting parameters
LR (baseline)	-
SVM	C=1.0 Kernel='RBF'
DT	Max Depth of Tree = 20
RF	Max Depth of Tree = 20 Number of Trees = 100
MLP	Output Dimension
FC - ReLU - Dropout (20%)	2048
FC - ReLU - Dropout (30%)	4096
FC - ReLU - Dropout (40%)	4096
FC - ReLU - Dropout (50%)	1024
FC - Softmax	9

The embedded feature $\mathbf{mean{-}num{-}1}^2 = (\bar{n}_1, \bar{n}_2, \ldots, \bar{n}_M)$ is computed by $\bar{n}_i = \frac{n_i}{\sum_{i=1}^{M} n_i}$.

D. Back-end Classifier

Given the embedding $\mathbf{mean{-}num{-}1}$, the baseline system uses Linear Regression for classifying them into nine genres of *Children's, Drama, Factual, Music, Sport, Weather, Comedy, Entertainment*, and *News*.

IV. CANDIDATE ARCHITECTURES

A. Candidate Statistics For Embedded Feature Extraction

With the baseline system proposed in Section III, only one sound event with the highest probability is used to tag on each 5-second audio segment. This may lead to a reduction in system performance in cases that certain sound events are dominant in a television programme. For example, as human speech is dominant in *News* and *Weather*, it is easy to misclassify between these two genres. This inspires us to evaluate whether multiple-sound-event tagging information (i.e. one segment is tagged by multiple sound events) is beneficial for representing each 5-second segment. Therefore, we propose to use k ($k = \{4, 6, 8, 10\}$) sound events which occupy top-k predicted probabilities to tag on each 5-second segment. We thus call $\mathbf{mean{-}num{-}k}$ as the embedded feature extracted when using the top-k sound events for tagging. Compute the embedding $\mathbf{mean{-}num{-}k}$ is same as $\mathbf{mean{-}num{-}1}$ proposed in the baseline system.

Furthermore, we evaluate whether predicted probabilities are beneficial for extracting embedded features. We refer to $\mathbf{mean{-}prob{-}k}$ as the embedded feature that is extracted from predicted probabilities. We firstly consider $\mathbf{p} = (p_1, p_2, \ldots, p_M)$ as *probability* vector, where M is the number of sound events defined in AudioSet dataset [14] and p_i is the total sum of predicted probabilities of the ith sound event detected in each programme. The embedding $\mathbf{mean{-}prob{-}k} = (\bar{p}_1, \bar{p}_2, \ldots, \bar{p}_M)$ is then computed by $\bar{p}_i = \frac{p_i}{\sum_{i=1}^{M} p_i}$.

B. Candidate Back-end Classifiers

In addition to Linear Regression (LR) used for classification in the baseline system, we further evaluate different back-end classification models. In particular, we apply both traditional machine learning models (Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF)) and a deep learning based model using Multilayer Perceptron (MLP) network architecture. The back-end classification models evaluated are configured as shown in Table II.

[2]Note that the number '1' in the embedding name is used to reflect that only one sound event is used to tag on one segment.

TABLE III
BBC TELEVISION PROGRAMME DATASET

Genres	Number of programmes
Children's	698
Drama	695
Factual	692
Music	670
News	700
Weather	654
Sport	663
Comedy	690
Entertainment	698
Total programmes	6,160

V. EVALUATION METHOD

A. Dataset

AudioSet: This is a large-scale dataset released by Google [14], in which there are around 2,084,320 10-second audio clips. This dataset contains a total of 527 sound event classes (including both natural sounds and human speeches) with annotation. Each 10-second audio clip may contain more than one type of sound events and there is no information of onset and offset for a certain sound event (i.e. weakly labelled dataset of sound events). This dataset is used for training the pre-trained CNN network as mentioned in our previous work [16].

BBC programmes: The dataset is collected by selecting programmes from eight different BBC TV channels (BBC One, BBC Two, BBC Four, CBBC, CBeebies, BBC News 24, BBC Parliament, and BBC World News) [21]. These programmes are from the nine main BBC genre categories [2] (*Children's, Drama, Factual, Music, Sport, Weather, Comedy, Entertainment* and *News*). Approximately 25 programmes per genre are collected for each month in the years 2019 and 2020, creating a dataset of 6,160 BBC programmes as shown in Table III. The audio is MP3-encoded at 128kbps joint stereo. To evaluate, we separate this dataset into 14-fold cross validation and report the final classification accuracy as an average over 14 folds.

B. Experimental setting

We construct the pre-trained CNN network by using Pytorch framework [16]. We use the following cross-entropy loss function during training of this network as

$$Loss_{EN}(\Theta) = -\frac{1}{N}\sum_{n=1}^{N} \mathbf{y_n} \log\{\hat{\mathbf{y}}_\mathbf{n}(\Theta)\} \quad (1)$$

defined over all parameters Θ, and N is the number of training clips. $\mathbf{y_n}$ and $\hat{\mathbf{y}}_\mathbf{n}$ denote ground truth and predicted output. The training is carried out for 100 epochs using Adam [22] for

TABLE IV
PERFORMANCE OF BACK-END CLASSIFICATION MODELS (LINEAR REGRESSION(LR), DECISION TREE (DT), SUPPORT VECTOR MACHINE (SVM), RANDOM FOREST(FR), AND MULTILAYER PERCEPTRON (MLP)) WITH DIFFERENT STATISTICS (AVERAGE ACCURACY OVER 14 FOLDS) .

Back-end Classifiers	mean-num-1 (%)	mean-prob-1 (%)	mean-num-4 (%)	mean-prob-4 (%)	mean-num-6 (%)	mean-prob-6 (%)	mean-num-8 (%)	mean-prob-8 (%)	mean-num-10 (%)	mean-prob-10 (%)
LR	56.7(**baseline**)	56.1	85.9	83.9	88.0	85.8	88.6	86.9	89.1	87.4
SVM	35.1	34.2	84.9	73.9	87.2	75.7	87.5	76.8	88.3	77.2
DT	61.5	60.2	79.8	78.9	79.0	79.5	80.2	79.3	80.5	79.4
RF	**69.4**	**68.5**	90.9	90.5	91.4	90.8	91.4	91.1	91.5	91.1
MLP	62.1	60.9	**92.6**	**91.6**	**93.2**	**92.1**	**93.5**	**92.2**	**93.7**	**92.4**

TABLE V
PERFORMANCE OF BACK-END CLASSIFICATION MODELS WITH THE COMBINED FEATURE (AVERAGE ACCURACY OVER 14 FOLDS)

Back-end Classifiers	mean-num-10 (%)	mean-prob-10 (%)	combined-feature (%)
LR	89.1	87.4	89.7
SVM	88.3	77.2	83.6
DT	80.5	79.4	80.4
RF	91.5	91.1	91.8
MLP	**93.7**	**92.4**	**93.6**

optimization. Regarding the back-end classification models, we use Scikit-learn toolbox [23] to implement traditional machine learning models such as Linear Regression (LR), Support Vector Machine (SVM), Decision Tree (DT), and Random Forest (RF). For Multilayer Perceptron (MLP), we also construct the network by the Pytorch framework. The loss function used for training the MLP based network is also cross-entropy (1). To enforce the MLP based network, we apply *mixup* data augmentation [24], [25] to audio embedded input features.

Regarding the evaluation metric used in this paper, if C is considered as the number of audio recordings of programmes which are correctly predicted, and the total number of audio recordings is T, the classification accuracy (Accuracy (%)) is the % ratio of C to T.

VI. RESULTS

Experimental results over both embedded features and all back-end classification models are presented in Table IV. As shown in Table IV, when the number of detected sound events used for classification increases, the accuracy is improved over all back-end classification models. It can be concluded that multiple-sound-event tagging is beneficial to extract embedding features rather than single-sound-event tagging. Comparing between the two types of embedding features, the sound event based embeddings perform better than predicted probability based embeddings over all back-end classifiers. Regarding back-end classification models evaluated, Random Forest and MLP based network outperform Linear Regression, Decision Tree and Support Vector Machine. The best score of 93.7% is obtained from MLP based network with $\mathbf{mean{-}num{-}10}$ embedding, significantly improving the baseline of $\mathbf{mean{-}num{-}1}$ embedding and back-end Linear Regression by approximately 37.0% .

The confusion matrix in Fig. 2 shows an average over 14 folds with $\mathbf{mean{-}num{-}10}$ embedding and back-end MLP model. It can be seen that wrong inference occurs among

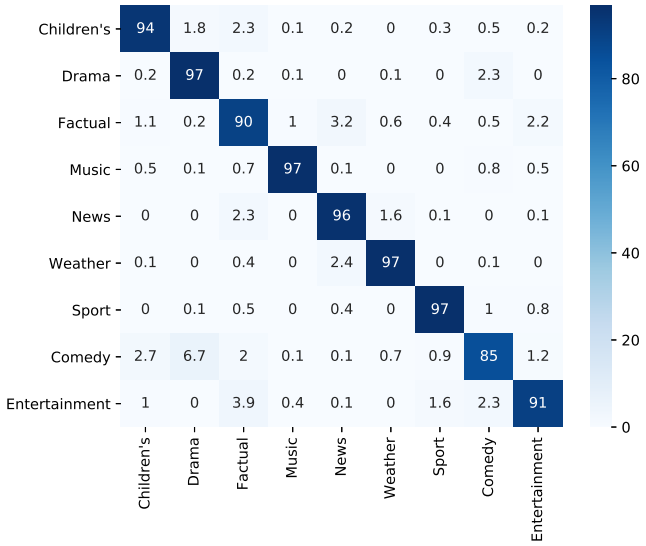

Fig. 2. Confusion matrix result (%) with $\mathbf{mean{-}num{-}10}$ embedding and back-end MLP (average Accuracy over 14 folds)

related programmes such as *Entertainment* and *Comedy*, or programmes of *News* and *Weather*. Meanwhile, *Music, Sport, Weather* and *Drama* achieve the best performance of aprroximately 97.0% among genres.

VII. FURTHER EXPERIMENTS

We also conduct further experiments to evaluate whether combination of both sound event based embedded feature and predicted probability based embedded feature, referred to as $\mathbf{combined{-}feature}$, can help to improve the performance. To this end, two embeddings are concatenated before feeding into the back-end classification models. As obtained results in Table V, $\mathbf{combined{-}feature}$ (a concatenation of $\mathbf{mean{-}prob{-}10}$ and $\mathbf{mean{-}num{-}10}$) helps Linear Regression and Random Forest improve the performance, but not effective for the other models.

As the duration of BBC programmes can be long, we evaluate whether a programme can be effectively detected with a reduced number of input segments, thus help to reduce the cost of inference process. In particular, 10% to 100% of the input segments are randomly selected from each programme for evaluation. We use $\mathbf{mean{-}num{-}10}$ embedding and all back-end classification models. As shown in Fig. 3, if 60% of segments or more are used, almost post-trained models' performance apart from Linear Regression is stable. Notably,

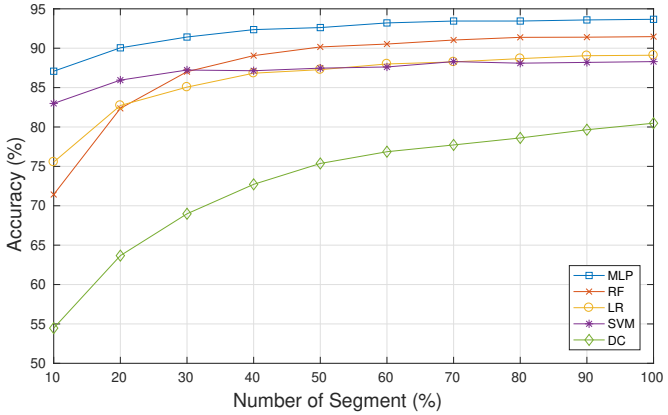

Fig. 3. Performance of all back-end classification models (Linear Regression (LR), Decision Tree (DT), Support Vector Machine (SVM), Random Forest (FR), and Multilayer Perceptron (MLP)) with reduced number of input segments and $\mathbf{mean{-}num{-}10}$ embedding (average of 14 folds)

MLP classifier achieves an average classification accuracy of 93.0% that potentially reduces 40% of time for the inference process.

VIII. Conclusion

We have explored a deep learning based framework for classifying BBC television programmes into nine genres that match the BBC genre categories. Our framework, which uses a log-mel spectrogram representation, a pre-trained CNN architecture for extracting embedded features, and a back-end Multilayer Perceptron classifier, achieved an average classification accuracy of 93.7% over 14-fold cross validation. In further work, we will evaluate whether the audio-based embedded features can be used to measure the similarity between BBC television programmes for the purpose of recommendations.

Acknowledgement

This work was funded by an EPSRC Impact Acceleration Account project EP/R511791/1, and carried out jointly by the University of Surrey and British Broadcasting Corporation (BBC).

References

[1] BBC, *2019 Annual Report*, https://www.bbc.co.uk.
[2] BBC, *BBC Programmes*, https://www.bbc.co.uk/programmes/genres/.
[3] X. Li-Qun and Y. Li, "Video classification using spatial-temporal features and pca," in *International Conference on Multimedia and Expo. (ICME)*, vol. 3, 2003, pp. III–485.
[4] M. Montagnuolo and A. Messina, "Automatic genre classification of tv programmes using gaussian mixture models and neural networks," in *18th International Workshop on Database and Expert Systems Applications (DEXA 2007)*, 2007, pp. 99–103.
[5] R. Glasberg, S. Schmiedeke, P. Kelm, and T. Sikora, "An automatic system for real-time video-genres detection using high-level-descriptors and a set of classifiers," in *IEEE International Symposium on Consumer Electronics*, 2008, pp. 1–4.
[6] J. Wu and M. Worring, "Efficient genre-specific semantic video indexing," *IEEE Transactions on Multimedia*, vol. 14, no. 2, pp. 291–302, 2012.
[7] D. Borth, J. Hees, M. Koch, A. Ulges, C. Schulze, T. Breuel, and R. Paredes, "Tubefiler: an automatic web video categorizer," in *Proceedings of the ACM Conference on Multimedia*, 2009, pp. 1111–1112.
[8] H. K. Ekenel, T. Semela, and R. Stiefelhagen, "Content-based video genre classification using multiple cues," in *Proceedings of the 3rd International Workshop on Automated Information Extraction in Media Production*, 2010, p. 21–26.
[9] Y. Zheng, L. Duan, Q. Tian, and J. S. Jin, "TV commercial classification by using multi-modal textual information," in *IEEE International Conference on Multimedia and Expo*, 2006, pp. 497–500.
[10] M. Doulaty, O. Saz-Torralba, W. Raymond, and T. Hain, "Automatic genre and show identification of broadcast media," in *INTERSPEECH*, 2016.
[11] M. Doulaty, O. Saz, and T. Hain, "Latent dirichlet allocation based organisation of broadcast media archives for deep neural network adaptation," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015, pp. 130–136.
[12] S. Kim, P. Georgiou, and S. S. Narayanan, "On-line genre classification of tv programs using audio content," *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 798–802, 2013.
[13] A. Gersho and R. M. Gray, *Vector Quantization and Signal compression*. Springer Science & Business Media, 2012, vol. 159.
[14] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. ICASSP*, 2017.
[15] B. McFee, R. Colin, L. Dawen, D. Ellis, M. Matt, B. Eric, and N. Oriol, "librosa: Audio and music signal analysis in python," in *Proceedings of The 14th Python in Science Conference*, 2015, pp. 18–25.
[16] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.
[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.
[18] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, 2015, pp. 448–456.
[19] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010.
[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
[21] BBC, *BBC Redux*, https://www.bbcredux.com/.
[22] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.
[23] 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.
[24] K. Xu, D. Feng, H. Mi, B. Zhu, D. Wang, L. Zhang, H. Cai, and S. Liu, "Mixup-based acoustic scene classification using multi-channel convolutional neural network," in *Pacific Rim Conference on Multimedia*, 2018, pp. 14–23.
[25] Y. Tokozume, Y. Ushiku, and T. Harada, "Learning from between-class examples for deep sound recognition," in *ICLR*, 2018.