Applied Deep Learning Coursework

*Note: This is a reproduction of the paper by Dieleman, Sander and Benjamin Schrauwen titled "End-to-end learning for music audio."

Vasudev Menon
Faculty of Engineering
University of Bristol
Bristol, United Kingdom
ee20947@bristol.ac.uk

***Abstract*—This paper examines the efficacy of end-to-end learning architectures in Music Information Retrieval (MIR). By replicating Dieleman and Schrauwen's 2014 study, it presents results from an automatic music tagging task across two different CNN architectures. It contains several comparisons between CNN architectures and training inputs while measuring the impact of Mel filter banks and FFT size on model performance.**

***Index Terms*—feature learning, MIR, spectrograms, waveform**

1 Introduction

Music information retrieval has traditionally relied on mid-level representations of music audio training, but recently there has been a push towards end-to-end architectures. These architectures connect the input to the desired output with minimal feature engineering effort. CNNs tend to work very well in this setting, because they consist of many layers of processing, all learned through the exact same objective function.

In the paper [2], it proposes the benefits for feature learning [4], as it allowed for the creation of models that required minimal prior knowledge. During the time of this paper, most research in the MIR community relied on feature learning based on mid-level representations of audio, such as spectrograms, which in and of itself is a form of prior knowledge. This approach, while effective, contrasts with the paradigms adopted within the domain of computer vision, where learning directly from raw data, specifically pixel-by-pixel analysis of images is considered the norm.

In the past few years, there is abundant research to incorporate end-to-end architectures in many different fields of deep learning. In the interest of the MIR space, there are music mastering, encoding and compression models being used to automate processes for artists and composers. This is lowering the bar to entry for people into an otherwise exclusive market.

In this paper, I delve deeper into the findings of Dieleman and Schrauwen et. al 2014. I investigate if it is possible to apply feature learning directly to the raw audio signals, without the prior knowledge required to use mid level representations of data like spectrograms. I use a convolutional neural network to solve an automatic tagging task, evaluating various inputs and network architectures. In Section 2, I discuss the current state of deep learning in the MIR community since the influential paper [2]. In Section 3, I describe the experiments done to replicate the results of Dieleman and Schrauwen. In Section 4, I present and discuss the results obtained from the study. In Section 5, I explore the improvements I made on my model and conclude in Section 6.

2 Related Work

2-A End-to-End Learning for Music Audio Tagging at Scale

This paper [13] from 2018 focused on comparing two models on a music audio tagging task at an unprecendented scale, testing on a private dataset of 1.2M songs. This comparison between a waveform model and a spectorgram model seems to be a clear successor to the research done by Sandler et al. It tries to tackle a similar problem, while providing a clear evidence-based approach to show that waveform-based models outperform spectrogram-based models with a large enough dataset. These experiments prove that previous assumptions of domain knowledge are only relevant when there is a lack of training data.

2-B Deep Neural Networks and End-to-End Learning for Audio Compression

This paper [15] from 2021 investigates end-to-end learned front ends in audio representation learning. It builds on the research by Sandler et al. by assessing the models ModNet and SincModNet, which incorporate a temporal modulation processing block. The authors propose promising results that substantiate modulation filtering as a valid method for music audio tagging, without the need for extensive musical domain knowledge.

2-C End-to-End Music Remastering System Using Self-Supervised and Adversarial Training

This paper [7] from 2022 discusses the development of an end-to-end self supervised music remastering system. The research tackles a more realistic application to the problem discussed by Sandler et al. The music remastering space has a high bar of entry due to the advanced technical knowledge required to to polish up songs to a certain high standard. The

end-to-end system proposed performs successfully to remaster an input audio into a target style in an adversarial manner.

3 Replication of Dieleman and Schrauwen's Experiments.

The paper "End-to-end learning for music audio" by Sander Dieleman and Benjamin Schrauwen [2], published in the 2014, made notable impact in the field of MIR. Their focus on applying feature learning directly to raw audio signals, as opposed to traditional mid-level representations like spectrograms, marked an important shift in the approach to audio analysis in MIR. It is characteristic of the broader trend in Machine Learning and Artificial Intelligence towards deep learning and end-to-end architectures.

3-A Dataset

The MagnaTagATune dataset [9] contains 25,863 29-second audio excerpts sampled at 16kHz from songs spanning 230 artists. Each clip comes labeled with up to 188 semantic tags describing musical characteristics like genres, instruments, moods, etc. Similar to the authors, I utilize this dataset for an automatic tagging task to evaluate my models' music labelling capabilities.

To rigorously test my model, I divided the data into training, validation, and test partitions. Parts 1-12 comprise training set, part 13 is used for validation monitoring during training, and the remaining parts 14-16 contain the untouched test set for final evaluation only. Due to the long-tailed distribution of the tags, I use the top 50 most popular tags for evaluation to ensure enough training data for each tag. This is consistently used as the structure for evaluating this dataset by researchers.[1] [1]

We use three different dataset collections. The one used for our base model replication of the CNN, is sampled at 12KHz to give audio samples of 29.1 second music clips. This is then further cut down into 10 subclips of 3 seconds for training. This data was normalised between -1 to 1 to allow for faster convergence of data.

For the extensions discussed in this paper, we use two types of spectrogram data. One has the same number of clips as the base model. The second one contains chunked down clips of 3.69 second to make it comparable to the model described in Won et al. 2020 [16]. This also allows us to make a comparison between our base model and improved model, since they both use chunked/sub clips of the inputs to train the data.

3-B BaseCNN Architecture

Following the structure of the paper, I have replicated the CNN architecture provided by the authors to make BaseCNN. As described in Figure 2, it consists of 6 total layers: two convolutional layers with 32 filters of length 8, alternating with max-pooling layers with pooling size 4, and two dense fully connected layers with 100 and 50 units respectively. I used rectified linear units in all layers except for the top layer, where I used sigmoidal units. I extended the architecture with some established deep learning principles. This includes batch normalisation [5] after each convolutional layer, scheduler and dropout. This extended architecture allows us to compare the model with more recent publications [1]. All convolutions and pooling operations are one-dimensional along the axis representing time.

[1] https://github.com/jongpillee/music_dataset_split/tree/master/MTAT

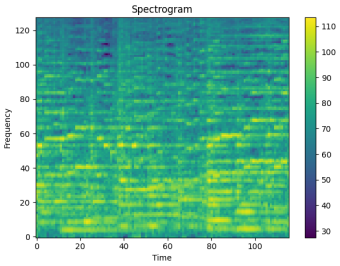

Figure 1: This spectorgram visually represents a spectrogram of size 1024 Fast Fourier Transform and 64 Mel filterbanks. It displays the spectrum of frequencies of the sample as it varies with time.

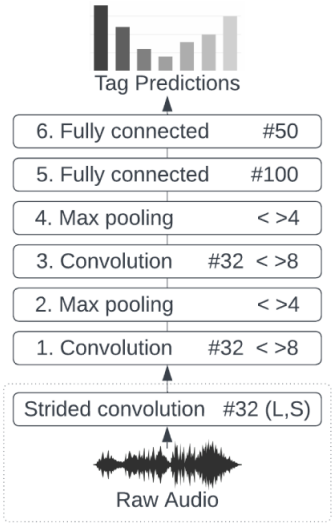

Figure 2: The convolutional neural network architecture replicated from Dieleman et al. 2014 [2]. The filter sizes and pooling sizes ($< >$), and number of units (#) are indicated. In this approach, I use a strided convolutional layer with variable length and stride (L,S) to explore various combinations of filters.

The network was trained using minibatch gradient descent. The data was sampled at 12KHz unlike the paper and then made into minibatches of 10 subclips with padding handled. To obtain the final prediction for the entire clip, the predictions from all these windows were averaged. The model's predictions were evaluated by calculating the area under the ROC curve (AUC) for each tag, and then averaging these scores across all 50 tags.

Additionally, the model was validated at regular intervals and tested after each run. The reported results are the parameters from the test set that achieved the best validation score. All experiments were run on University of Bristol's Blue Crystal 4 with GPU acceleration.

3-C Implementation Details

The datasets were split into the training, validation and test splits, to maintain similarity with other research done on this dataset. I tested out the direct replication of the model described in Sandler et al. 2014, as described in the CNN architecture. Since there was no mention of the number of filters in the strided convolution, I used 32, to keep consistent with the other layers.

This gave me a base accuracy of 74% without any hypertuning. Once the scheduler, batch normalisation and dropout were added it shot up my accuracy to 85% after hyperparameter tuning. The paper mentions that dropout doesn't affect their accuracy, but in our case it does. This is possibly due to a different sampling rate to get the data from the MagnaTagATune dataset.

L2 regularisation was added on top of this maintain to slightly improve accuracy. Though some papers mention that batch norm and regularisation don't work together [8], empirically I got better results with them both turned on.

In terms of optimizers, Adam, Rmsprop, Adagrad, SGD were used and SGD tended to perform better over more tests so that was used for further testing.

Since spectrograms are generally preferred over raw audio, it was an obvious next choice. I tested multiple different architectures including a ResNet, CRNN, vanilla CNN and finally decided on the ChunkResCNN. This follows the architecture of the Short-chunk CNN with residual connections, which has the highest ROC-AUC score on the MagnaTagATune dataset across all models tested on this dataset [12]. This model was tested on 6 different kinds of spectrogram datasets to get the best accuracy.

Once all the testing was complete, I modularised my codebase, allowing for easy reproducibility of the architecture, hyperparameter tuning and training of the model.

4 Results and Insights

During the creation of both networks discussed in this paper, many variations of implementation were tested, but the results in this section are those which empirically gave the best results over 600+ runs.

4-A Quantitative Results

The quantitative results from the experiments run are shown below.

It is clear from the results that the accuracy of the BaseCNN tends to increase as the length and stride of the filter layers decrease. This could imply that the smaller filters tend to capture more relevant features from our data [3]. Since the data is essentially just temporal, the smaller lengths may allow for a more nuanced learning of temporal transitions, therefore increasing its tagging accuracy. The decrease in performance for larger filter lengths and strides could also mean that BaseCNN might work poorly with a larger context window. Even lower filter lengths and strides give us diminishing returns as shown in the table. It is also interesting to note that as the length and stride increase, so does the overfitting of the model. In Figure 3, the purple curves (1024, 1024) have a larger generalisation gap as compared to the green curves (256, 256). This could be the case because the larger length and stride cause aggressive downsampling, leading to a loss of fine grained detail, making the model more prone to overfitting as it can't generalise well to new data [11].

length	stride	AUC (raw audio)
1024	1024	0.8044
1024	512	0.8188
512	512	0.8265
512	256	0.8274
256	256	0.8501

Table 1: Results for the audio tagging task on MagnaTagATune dataset using BaseCNN. It shows the effect of different combinations of lengths and strides for our strided convolution. I report AUC scores on the test set.

Model	AUC Score
BaseCNN	0.8501
BaseCNNNonInitialised	0.8455
CRNN	0.8854
ChunkResCNN	0.9043

Table 2: Results for the audio tagging task on MagnaTagATune dataset using ChunkResCNN. It shows the effect of different combinations of Mel filterbanks and Size of Fast Fourier Transforms on accuracy. I report AUC scores on the test set. The bottom two values are on the spectrograms of 3.69sec chunks.

Table 2 shows a comparison between BaseCNN with and without kaiming initialisation, CRNN and the ChunkResCNN. This was done to understand the effect of kaiming initialisation on our base model since it uses multiple RELU activation layers. Though not a substantial improvement in accuracy, when looking at its associated graphs, it caused the model to converge faster and maintain accuracy with less fluctuations in the final epochs. We compare this BaseCNN to a spectrogram based CRNN and ChunkResCNN discused in Section 5-A. Though the spectrogram based models perform better the accuracy of BaseCNN still maintains a very high accuracy.

4-B Training Curves

4-C Qualitative Results

The model performs very well (AUC $>$ 0.9) on the tags 'choir', 'opera' and 'rock'. This is possibly because these kinds of music are very distinctive and has substantially consistent characteristics across varieties of songs. Even to

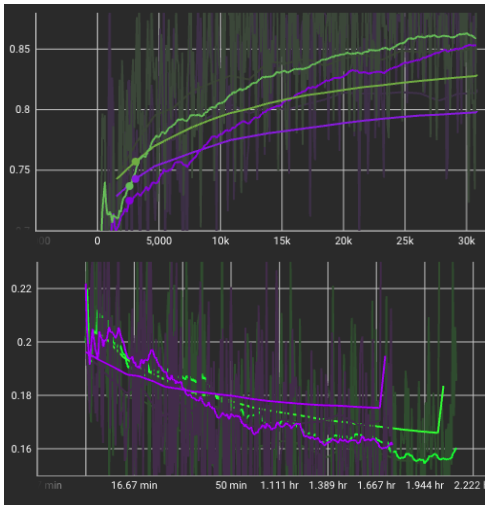

Figure 3: The above curves are the training and loss curves for BaseCNN corresponding to the first and last set of lengths and strides mentioned in Table 1. The purple curves represent L,S = 1024,1024 and the green curves represent L,S = 256,256.

the blind ear these genres could be picked up very easily, because they tend to follow consistent patterns followed by most artists that make the genre of music. Choirs have multiple voices singing in unison, opera singers are trained to use a vibrato not found in any other genre and rock music tends to have a strong rhythm, usually accompanied by distortion from the electric guitar.

On the flip side, the tags 'no beat', 'no singer' and 'no piano' tends to do the worst on this model (AUC < 0.7). This is usually a struggle for most models because it is a detection of absence vs presence. This means that for a model to be able to predict that a sound is 'not piano' it requires to interpret every single sound except the piano, which proves to be an obvious challenge when you put it into scale how much music already exists and how the music industry is still growing exponentially as more far reaching technologies emerge. There is a lack of literature to cover this because unlike positive features, there are no distinct nuances or patterns for the model to learn [10]. This raises a more issues when you try to train the model for such tags, because the model might confuse incidental features due to the lack of certainty. It is also hard to produce negative samples for training, because it is hard to quantify what that means in reality.

5 Improvements

5-A ChunkResCNN

For the extension, I chose to implement a completely new architecture to use on spectrograms. Spectrograms are the most widely used form of input for MIR tasks. The reasons for this are many fold. They afford noise reduction and feature enhancement using techniques like the Mel scale to attain more perceptually relevant features to humans. There is also the added benefit being able to use them as images. Allowing us to use powerful and more established image processing techniques. This equates to better feature extraction and pattern recognition.

The biggest advantage offered by spectrograms is the dimensionality reduction which reduces computational load on models. After experimenting with raw audio inputs, mel-scaled spectrograms empirically outperformed, providing a significant 5% boost in AUC score (0.9043) for this audio tagging task on a specific run for 30 epochs.

The ChunkResCNN described in this paper follows the architecture described in Figure 4.

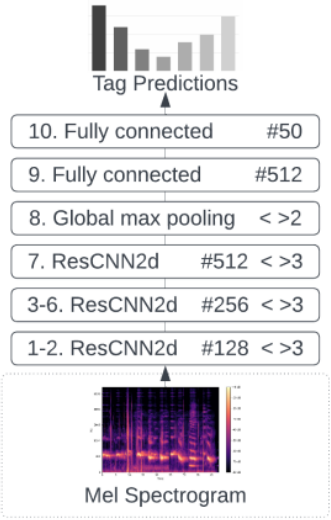

Figure 4: The neural network architecture replicated from Won et al. 2020 [16]. The filter sizes and pooling sizes ($<$ $>$), and number of units (#) are indicated. Each ResCNN2d layer is made up of two convolutional layers with a residual connection

5-B Effect of different spectrogram types

This model performs better in than BaseCNN across all trials. This is probably due to the factors mentioned above. This model was trained on 6 different sets of spectograms to test the effect of the Mel Bins and Fast Fourier Transform on accuracy. Table 3 shows the accuracy of the different spectrograms created. The chunked spectrograms contain short 3.69s audio segments, which captures potent timbral and temporal audio features.

Table 3 confirms that the short chunked spectrogram performs the best. It is also empirically proves that having a larger Fast Fourier Transform and more Mel filterbanks improves accuracy. The above parameters while creating the spectrogram allow for greater temporal resolution and increased feature sensitivity, subsequently promoting better performance of the model [6].

It is important to note that there is a noticeable trade-off on computational load vs accuracy in the experiment below. Though the chunked spectrograms performed better, they took

substantially longer to train due to the sixfold increase in data volume.

No. of Mel filterbanks	Size of FFT	AUC (spectogram)
1024	128	0.8810
1024	64	0.8771
512	128	0.8283
512	64	0.7949
1024	128	0.8955
512	128	0.8707

Table 3: Results for the audio tagging task on MagnaTagATune dataset using ChunkResCNN. It shows the effect of different combinations of Mel filterbanks and Size of Fast Fourier Transforms on accuracy. I report AUC scores on the test set. The bottom two values are on the spectrograms of 3.69sec chunks.

The ChunkResCNN is a deeper neural network than the BaseCNN, so it is more prone to the vanishing gradient problem. This is why the residual connections are very useful in this network, ensuring that the additional layers don't hinder performance. This was proven to be the case when our model was tested over larger epochs to test for over fitting.

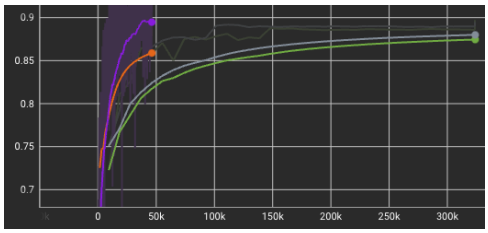

Figure 5: The above curves depict the vanishing gradient problem faced by the ChunkResCNN model trained on different spectrograms. The bottom two depict the test and train curves on the chunked spectrogram and the top two depict the test and train curves on the basic spectrogram.

From figure 5 we see that the ChunkResCNN tends to reach capacity very quickly. After epoch 8, the curve flattened out across training runs with similar hyperparameters. This suggests that the model reaches convergence very quickly and settles there. There does seem to be significant overfitting on the basic spectrogram, while there is a negligible gap between the curves of the chunked spectrogram. This supports the notion that smaller chunks of audio tend to improve training of such models.

6 Conclusion and Future Work

In conclusion, I have explored how the results of Sandler et al. hold up after almost a decade at the time of writing. I have compared the multiple different types of architectures and data types to empirically test which performs the best. I quantitatively and qualitatively categorised outcomes from my experiments. Finally, I discussed related works in the MIR space that strongly support the arguments made by Sandler et al.

In future work I will investigate the effects of data augmentation techniques and attention mechanisms in music information retrieval. This may increase the robustness and performance of my model as larger datasets seem to improve performance significantly on waveform based models [13]. I would further investigate the effects of a smaller filter size in both the waveform and spectrogram based models to test the vanishing gradient of this model. I will also implement bayesian hyperparameter optimisation on my model to attain the best possible accuracy while significantly improve efficiency of hyperparameter tuning and training [17] [14].

References

[1] Niraj Anil Babar, J. Sumanth, Noel Alben, Neeluru Sai Deepika, S.N.R Ajey, and M. Pramod. Active 2d sound source localization using a sample dcnn architecture encoding. *2022 2nd International Conference on Intelligent Technologies (CONIT)*, pages 1–6, 2022.

[2] Sander Dieleman and Benjamin Schrauwen. End-to-end learning for music audio. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6964–6968, 2014.

[3] Vikas Ghai, Kamal Sharma, Zainul Hasanali, Sara Shimko, August Stuart, Jason Liao, and Elliot Epner. A retrospective study analyzing activity of ofatumumab in blastic variant mantle cell lymphoma. *Journal of Clinical Oncology*, 30:e18500–e18500, 05 2012.

[4] Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[5] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ArXiv*, abs/1502.03167, 2015.

[6] Hemant Kathania, Syed Shahnawazuddin, Waquar Ahmad, and Nagaraj Adiga. Role of linear, mel and inverse-mel filterbanks in automatic recognition of speech from high-pitched speakers. *Circuits, Systems, and Signal Processing*, 38, 10 2019.

[7] Junghyun Koo, Seungryeol Paik, and Kyogu Lee. End-to-end music remastering system using self-supervised and adversarial training, 2022.

[8] T. V. Laarhoven. L2 regularization versus batch and weight normalization. *ArXiv*, abs/1706.05350, 2017.

[9] Edith LM Law, Kerry West, Michael I Mandel, Mert Bay, and J Stephen Downie. Evaluation of algorithms using games: The case of music tagging. In *Proceedings of the 10th International Society for Music Information Retrieval Conference*, 2009.

[10] Gaurav Kumar Nayak, Konda Reddy Mopuri, Saksham Jain, and Anirban Chakraborty. Mining data impressions from deep models as substitute for the unavailable training data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):8465–8481, 2022.

[11] Cassandra Newell and LeChelle Nelson. Over-the-counter medication prescribing in a pediatric emergency department: Health records review. *Journal of Emergency Nursing*, 48, 10 2021.

[12] Papers With Code. MagnaTagATune.

[13] Jordi Pons, Oriol Nieto, Matthew Prockup, Erik M. Schmidt, Andreas F. Ehmann, and Xavier Serra. End-to-end learning for music audio tagging at scale. *CoRR*, abs/1711.02520, 2017.

[14] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[15] Cyrus Vahidi, Charalampos Saitis, and György Fazekas. A modulation front-end for music audio tagging, 2021.

[16] Minz Won, Andres Ferraro, Dmitry Bogdanov, and Xavier Serra. Evaluation of cnn-based automatic music tagging models, 2020.

[17] Jia Wu, Xiu-Yun Chen, Hao Zhang, Li-Dong Xiong, Hang Lei, and Si-Hao Deng. Hyperparameter optimization for machine learning models based on bayesian optimizationb. *Journal of Electronic Science and Technology*, 17(1):26–40, 2019.