Journal of Computer Science

Original Research Paper

Efficient Music Auto-Tagging with Convolutional Neural Networks

Shaleen Bengani, S. Vadivel and J. Angel Arul Jothi

Department of Computer Science, Birla Institute of Technology and Science Pilani Dubai Campus, Dubai, UAE

Article history
Received: 27-04-2019
Revised: 15-07-2019
Accepted: 23-08-2019

Corresponding Authors:
S. Vadivel
Department of Computer Science, Birla Institute of Technology and Science Pilani Dubai Campus, Dubai, UAE
Email: vadivel@dubai.bits-pilani.ac.in

Abstract: Technology is revolutionizing the way in which music is distributed and consumed. As a result, millions of songs are instantly available to millions of people, on the Internet. This has created the need for novel music search and discovery services. Music is often searched using descriptive keywords, or tags, based on the content of the song. Hence, one very important task in achieving a great music search engine is automatic tagging of music. Currently, deep learning techniques using convolutional neural networks produce state- of-the-art results for this task. Several deep learning algorithms are able to achieve good results but at the cost of efficiency. As neural networks get deeper, the cost of computation grows exponentially. In this paper, we present a deep learning-based ensemble method that achieves near state-of-the-art performance on the music auto-tagging task. Our method is significantly more efficient in terms of computation time and disk space. This opens up the option of using our proposed model directly on a mobile device.

Keywords: Deep Learning, Convolutional Neural Network, Music Auto-Tagging, Mel Spectrogram

Introduction

About 15 years ago, most people obtained the music they listened to from music CDs. With a majority of the world now having internet, this has changed. The advent of music streaming services such as Spotify, Apple Music and Pandora, people have instant access to millions of songs. Spotify, the most popular music streaming service in the world, for instance, has over 40 million songs and more than 180 million users. Apple Music and Pandora, too, have about the same number of songs in their library. So, these companies often compete on features such as music discovery and recommendation.

Music is often described using special keywords called tags. Tags convey information such as instruments (e.g., piano, guitar, drums), genre (e.g., rock, pop, classical, indie), mood (e.g., happy, sad, upbeat). Manually adding tags to millions of songs can be very expensive and time consuming. This has made automatic tagging of music a very important task in the field of music information retrieval. Music auto-tagging is a classification problem of predicting music tags using audio signals.

As deep learning was not that popular in 2011, majority of the papers surveyed involved a feature extractor with manually designed features followed by a classifier. This also required the researchers to have a fair amount of domain-level knowledge to understand what kind of features could satisfactorily describe acoustic properties (Fu *et al*., 2011). Whereas in recent years, deep neural networks have been successfully used to annotate music. However, deep learning models are more often than not computationally very expensive and occupy a lot of disk space. Hence, they cannot exist on mobile device. The following subsection details the related work in music auto-tagging.

Related Work

Convolutional Neural Networks (CNNs) perform really well on pattern recognition tasks because of their ability to learn spatially invariant features. Naturally, several researchers have also used CNNs for music auto tagging.

Dieleman and Schrauwen (2014) used a two layer CNN with Mel spectrogram as well as raw audio as input. They experimented with various convolutional kernel sizes

Science Publications

© 2019 Shaleen Bengani, S. Vadivel and J. Angel Arul Jothi. This open access article is distributed under a Creative Commons Attribution (CC-BY) 3.0 license.

and strides. Liu and Yang (2016) used a fully convolutional neural network (FCN) in a weakly supervised method to annotate each time frame of an audio clip from just the clip level information. Choi *et al*. (2016) experimented with fully convolutional networks of various depths on the MagnaTagATune dataset and Million Song Dataset (Bertin-Mahieux *et al*., 2011). They find that deeper models work better on larger datasets such as the Million Song Dataset. Lee *et al*. (2017) proposed an end-to-end system in which a 9 layer 1D convolutional network operates on small samples of raw audio. Kim *et al*. (2018) used modern CNN architectures such as residual neural networks and squeeze and excitation networks (He *et al*., 2016; Hu *et al*., 2017).

Challenges

While methods proposed by Dieleman and Schrauwen (2014); Liu and Yang (2016); Choi *et al*. (2016) are somewhat computationally efficient, they perform poorly when compared to the state-of-the-art. The methods proposed by Kim *et al*. (2018) and Lee *et al*. (2017) produce excellent results, but they use very deep convolutional neural networks. These models are compute intensive and have a large memory footprint.

Contribution

In this study we propose an efficient method that solves the problem of music auto-tagging. The proposed method uses an ensemble of lightweight and efficient deep learning models. Our method is more efficient as it only uses a few megabytes size and can annotate music in less than a second. Thus, the model can be used in mobile devices. The ensemble comprises of three models, two that work on Mel spectrograms, and one that works on raw audio samples. We use the MagnaTagATune dataset to train and evaluate the models.

Motivation

While music streaming services are gaining popularity, a lot of people still download music from various sources on the internet. These files usually lack audio tags. In such cases, on device music auto tagging has several benefits. The organisation providing the service saves on cloud infrastructure costs. Transferring audio files over the internet is an added cost overhead. It also consumes a lot of bandwidth. On device auto tagging also allows the user to tag songs in the absence of a network connection. Finally, there are no privacy concerns as the files never leave the user's device.

This paper is structured as follows: In the next section, we discuss the dataset used. Later, we describe the proposed algorithm. After that, we present the experiments followed by the results obtained. Then we give the discussion and finally we conclude the paper.

Material

Dataset

We have used the MagnaTagATune (MTAT) dataset for our experiments. It is one of the most popular publicly available datasets for music annotations. MTAT has 29. 1s clips for 25, 863 songs. The data was obtained from independent artists and some clips belong to the same song. The dataset includes 188 tags across various categories such as instruments used, genre and mood. Figure 1 shows the frequency of tags in the MagnaTagATune dataset. It can be observed from Fig. 1 that the tags are not evenly distributed. Tags beyond the top 50 by frequency have very few samples. It is significantly harder to train any deep learning model to classify those labels with fewer samples. Moreover, all existing works have dealt with only the top 50 labels. Hence, in this research work we use only the top 50 most frequently occurring tags for our experiments. MTAT comes partitioned into 16 sets (0 to f). We combine and then shuffle all of them. We split the data into training, validation and test sets with a 13:1:2 ratios.

Methods

The proposed method is an ensemble of three convolutional neural networks named as Model 1, Model 2 and Model 3. The first two networks (Model 1 and Model 2) work on Mel spectrograms while the third model works on raw audio samples. Figure 2 illustrates the proposed model.

Acquiring Mel Spectrogram and Pre-Processing

For the two models working on Mel spectrograms, the raw audio samples are converted to Mel spectrograms with 128 frequency bins and 20 pixels per second. This gives us spectrograms of shape 128×592. The initial intuition behind using spectrograms was to convert an audio classification problem into an image classification one since convolutional neural networks have performed extremely well on a wide range of image-based tasks. Moreover, it helps in input dimensionality reduction (128×592 = 75,776 vs 4,65,000). We arrived at this particular shape (128×592) after comparing its performance with higher resolution spectrograms. The higher resolution spectrograms did not provide any improvement in the AUC score. The spectrograms are normalized by subtracting the mean and then dividing by the standard deviation. The spectrograms are normalized because CNNs converge faster when trained on normalized input.

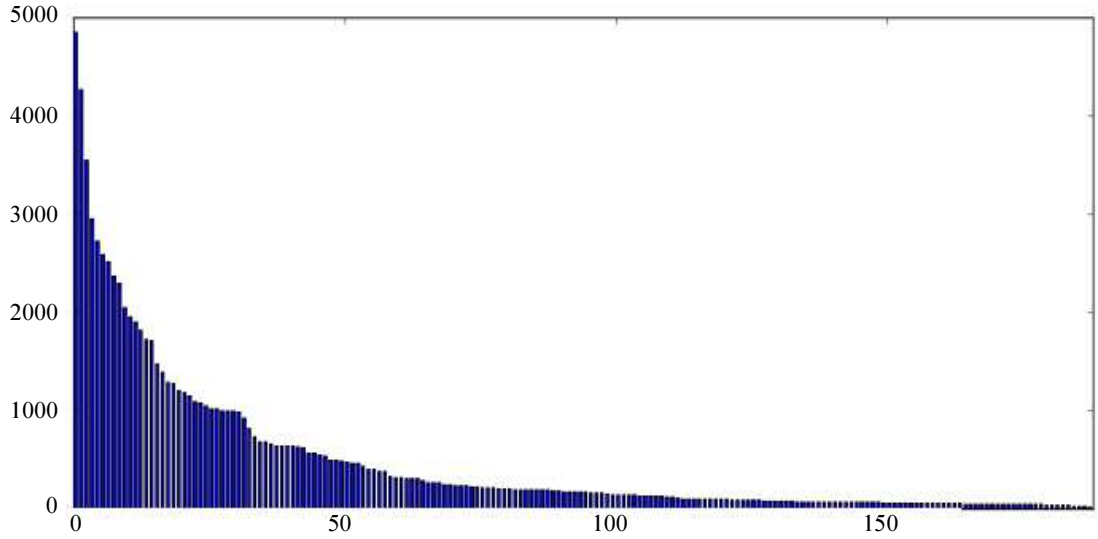

Fig. 1: Frequency of tags in the MagnaTagATune dataset

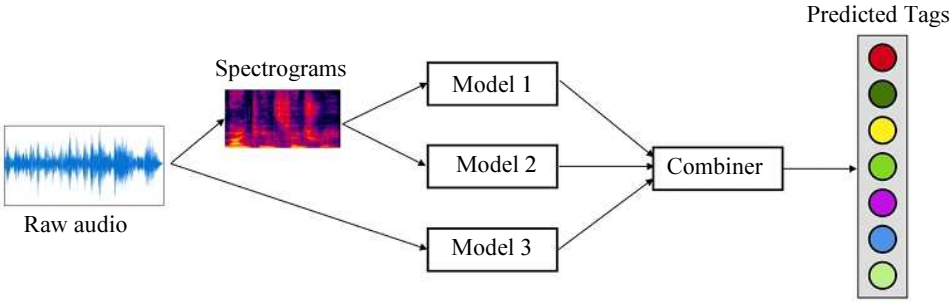

Fig. 2: Proposed method

Model 1: 1D Convolutions on Mel Spectrograms

In this model, we perform 1D convolutions on the Mel spectrogram. While called a 1D convolution, the underlying operation is a 2D convolution where the height of the filter in the first convolution layer is equal to the number of frequency bins in the spectrogram, i.e., 128. So, the convolution operation moves along the time axis. The following layer will have a height of 1, so the regular 1D convolution operation can be applied from there on. The goal of such a model is to establish invariance along the time axis as a particular music feature can occur at any time frame but will have a defined frequency pattern. Furthermore, 1D convolutional layers are less demanding computationally as the number of operations performed, as well as the number of parameters, is significantly less than multidimensional convolutional layers. Table 1 describes the architecture of the Model 1. While computationally efficient, fully connected layers occupy a lot of memory because of the large number of connections. However, in this case, we can get away with them by using 1D convolution instead of fully connected layers.

Model 2: 2D Convolutions on Mel Spectrograms

In 1D CNNs, the number of parameters doesn't grow by a lot with increasing depth. As a result, deep 1D CNNs tend to overfit quickly in the case of medium sized datasets like MTAT. We use convolutional layers with small sized kernels for this task. Just like in MobileNets (Howard *et al*., 2017), we apply depth wise separable convolutions to reduce the number of parameters and multiplication-addition operations in the network. Table 2 describes the architecture of the Model 2.

Model 3: 1D Convolutions on Raw Audio

In this method, we sample the raw audio at a rate of 16 KHz. For a 29.1s clip, this gives 465,000 samples. We normalise the training data by subtracting the mean and dividing by standard deviation. We also make use of dropout (Hinton *et al*., 2012) and batch normalization (Ioffe and Szegedy, 2015) layers to reduce bias and speed up training. The architecture of the network is described in Table 3.

Table 1: Configuration of model 1

Mel spectrogram (Input: 128×592×1)
Conv 128×4×256 (ReLU, Output: 1×589×256)
Reshape (Target Shape: 589×256)
Max Pooling 2 (Output: 294×256)
Conv 4×256 (ReLU, Output: 291×256)
Batch Normalization
Max Pooling 2 (Output: 145×256)
Conv 4×384 (ReLU, Output: 142×384)
Max Pooling 2 (Output 71×384)
Batch Normalization
Conv 71×50 (Sigmoid, Output: 50×1)

Table 2: Configuration of model 2

Mel spectrogram (Input: 128×592×1)
Conv 3×1×128 (ReLU, Output: 128×592×128)
Conv 1×3×128 (ReLU, Output: 128×592×128)
Max Pooling 4×4 (Output: 32×148×128)
Conv 3×1×256 (ReLU, Output: 32×148×256)
Conv 1×3×256 (ReLU, Output: 32×148×256)
Max Pooling 4×4 (Output: 8×37×256)
Conv 3×1×256 (ReLU, Output: 8×37×256)
Conv 1×3×256 (ReLU, Output: 8×37×256)
Max Pooling 2×2 (Output: 4×18×256)
Conv 3×1×512 (ReLU, Output: 2×18×512, Valid padding)
Conv 1×3×512 (ReLU, Output: 2 x 16×512, Valid padding)
Max Pooling 2×2 (Output: 1×8×512)
Conv 1×8×50 (Sigmoid, Output: 1×1×50)
Reshape (Output: 50×1)

Table 3: Configuration of model 3

Raw Audio (Input: 465600×1)
Conv 5×64 (ReLU, Output: 465596×64)
Max Pooling 16 (Output: 29099×64)
Conv 3×128 (ReLU, Output: 29097×128)
Conv 3×128 (ReLU, Output: 29095×128)
Max Pooling 4 (Output: 7273×128)
Batch Normalization
Dropout 0.2
Conv 3×128 (ReLU, Output: 7271×128)
Conv 3×128 (ReLU, Output: 7269×128)
Max Pooling 4 (Output: 1817×128)
Batch Normalization
Dropout 0.2
Conv 3×256 (ReLU, Output: 1815×256)
Conv 3×256 (ReLU, Output: 1813×256)
Global Max Pooling (Output: 256)
Fully Connected 1024 (ReLU, Output: 1024)
Fully Connected 50 (Sigmoid, Output: 50

Combiner

The combiner takes the outputs of models 1, 2 and 3 and produces the final output. It's a simple fully connected neural network with two hidden layers. The structure of the combiner is described in Table 4.

Experiments and Results

Implementation

All the models are implemented using the Keras API (https://github.com/fchollet/keras) for Tensorflow (Abadi *et al*., 2015). The models were trained on an Nvidia Tesla V100 GPU. We have used binary cross entropy loss for all the models (Model 1, Model 2, Model 3) with Adam optimizer.

Evaluation Metrics

In order to measure the performance of the proposed models, we use the area under receiver-operator curve (AUC) metric. The AUC range lies between 0.5 to 1.0 where 0.5 denotes a bad classifier model and 1.0 denotes a good classifier model. AUC is more robust to changes in test set distribution as it depends on true positive and false positive rates. Moreover, majority of the previous works have used AUC to evaluate their model's performance. So, in this study we use AUC to make a fair and easy comparison. We also tabulate the time taken for the model to predict tags for 1000 songs.

Experiments Conducted

In order to evaluate the performance of the proposed models in music auto-tagging we conducted the following experiments.

Evaluation of Model 1: In this experiment, Model 1 is trained and tested using Mel spectrograms and the AUC is calculated.

Evaluation of Model 2: In this experiment, Model 2 is trained and tested using Mel spectrograms and the AUC is calculated.

Evaluation of Model 3: In this experiment, Model 3 is trained and tested using raw audio samples and the AUC is calculated.

Evaluation of ensemble model: In this experiment, the individual models mentioned above like Model 1, Model 2 and Model 3 are combined using the combiner and the combined model as shown in Table 4 and the AUC is calculated. This experiment helps us to understand the importance of combining the models to achieve better performance in contrast to the performance of the individual models.

Evaluation of Ensemble Model Against Other Methods

In this experiment, the ensemble model is trained and tested and the AUC is calculated. This experiment helps us to understand the importance of the combining the models to achieve better performance in contrast to the performance of the other proposed models in the literature.

Table 5 shows the performance comparison of Model 1, 2 and 3 individually, as well as that of various ensemble combinations. Table 6 shows the performance comparison of other methods with the proposed ensemble model.

Table 4: Configuration of combiner

Model 1 output	Model 2 output	Model 3 output
Concatenate	Model	Outputs
Fully	Connected	256 (ReLU)
Fully	Connected	512 (ReLU)
Fully	Connected	50 (Sigmoid)

Table 5: Performance comparison of proposed models

Model	AUC	Time(s)
Model 1	0.857	0.48
Model 2	0.891	1.76
Model 3	0.783	9.03
Model 1+Model 2	0.894	2.07
Model 1+Model 3	0.883	9.24
Model 2+Model 3	0.905	9.78
Model 1+Model 2+Model 3	0.910	10.61

Discussion

Performance Analysis of Individual and Ensemble Models

From Table 5 it is clear that none of the individual models could achieve good performance. Model 1, Model 2 and Model 3 have AUC values of 0.857, 0.891 and 0.783 respectively. It is evident from the table that if two models are combined, then the combined performance of the models is greater than the individual performance of each models. For example, from Table 5 it can be seen that the combined performance of Model 1 and Model 2 is AUC = 0.894 which is greater than the performance of Model 1 (AUC = 0.857) and Model 2 (AUC = 0.891) separately. The combined performance of Model 1 and Model 3 is AUC = 0.883 which is greater than the performance of Model 1 (AUC = 0.857) and Model 3 (AUC = 0.783) separately. Also, the combined performance of Model 2 and Model 3 is AUC = 0.905 which is greater than the performance of Model 2 (AUC = 0.891) and Model 3 (AUC = 0.783) separately. Finally combining all three models yields the best AUC value of 0.910. Values in the 0.88 to 0.89 range haven't been considered to be that good because previous works from even 5 years ago have been able to achieve similar performance. 0.910 is very close to the current state of the art which is 0.911.

It can be seen from the results that among the two models (Model 1 and Model 2) that operate on the Mel spectrograms, Model 2 has a better AUC score. This is because Model 2 uses the 2D convolution of the Mel spectrograms and has learnt better features. Hence, combining Model 2 with other models like Model 1 or Model 3 provides a better AUC score. Finally, it can be noted that combining the models that works on spectrogram inputs and the raw audio increases the performance. This means that models working on spectrograms and raw audio learn different kinds of features.

Table 6: Performance comparison of proposed model with other state-of-the-art methods

Model	AUC
Dieleman and Schrauwen (2014)	0.881
Liu and Yang (2016)	0.896
FCN-4 (Choi *et al*., 2016)	0.894
Multi-D CNN (Pons *et al*., 2017)	0.893
Sample-level CNN (Lee *et al*., 2017)	0.905
ReSE-2-multisample (Kim *et al*., 2018)	0.911
Model 1+Model 2+Model 3	0.910

Performance Analysis of Ensemble Model Against other Methods

Table 6 shows the AUC value of the proposed ensemble model along with few other models discussed in the related work section of this paper. It can be observed that among other state-of-the-art methods, the proposed model is better than most of the other models (Dieleman and Schrauwen, 2014; Liu and Yang, 2016; Choi *et al*., 2016; Pons *et al*., 2017; Lee *et al*., 2017). The performance of the proposed model is in par with the performance of the model proposed by Kim *et al*. (2018) (AUC = 0.910 and AUC = 0.911 respectively). Thus the results show that the proposed ensemble model works in par with the model proposed by Kim *et al*. (2018) and other state-of-the-art models.

Performance Analysis of Model Size and Time

The sizes of Model 1, 2 and 3 and the combiner are 13.59 MB, 8.14 MB, 3.09 MB and 784 KB respectively. Therefore, the overall model has a size of just 25.60 MB. Moreover, on a Tesla V100 GPU, the model can predict tags for 1000 songs in just 10.61s. If a little compromise on accuracy is acceptable, using just Model 1 and Model 2, the same task can be performed in a just 2.07 seconds. That is just 2ms per song, making it feasible to deploy the model on mobile devices.

Conclusion

This research work presented an ensemble of three convolutional neural networks that automatically tags music clips with the goal of being accurate yet computationally efficient. From our experiments with different ensemble combinations, we inferred that CNNs working on Mel spectrograms and raw audio tend to learn different kinds of acoustic features. Experimental results show that the proposed ensemble model performs on par with the state-of-the-art methods.

Funding Information

The Authors declare that there are no funding sources for this research work.

Author's Contributions

Shaleen Bengani: Contributed in identifying the methods, performed experiments, analyzed the results and drafted the paper.

S. Vadivel: Contributed in overall monitoring of the research work, identifying the methods, analysis of results, interpretation of the results and reviewing the paper.

J. Angel Arul Jothi: Contributed to this research by critically reviewing the literature, methodology, experiments, results and the manuscript for significant intellectual content.

Ethics

This work is original and not published elsewhere. The authors confirm that they have read and approved the manuscript and there is no conflict of interest. Also, the authors declare that there are no ethical issues associated with this research work.

References

Abadi, M., A. Agarwal, P. Barham, E. Brevdo and Z. Chen *et al*., 2015. TensorFlow: Large-scale machine learning on heterogeneous distributed systems.

Bertin-Mahieux, T., D.P. Ellis, B. Whitman and P. Lamere, 2011. The million song dataset. Proceedings of the 12th International Conference on Music Information Retrieval, Oct. 24-28, Miami, Florida, USA.

Choi, K., G. Fazekas and M.B. Sandler, 2016. Automatic tagging using deep convolutional neural networks. Proceedings of the International Society for Music Information Retrieval Conference, (IRC' 16), New York, pp: 805-811.

Dieleman, S. and B. Schrauwen, 2014. End-to-end learning for music audio. Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, May 4-9, IEEE Xplore Press, Florence, Italy, pp: 6964-6968. DOI: 10.1109/ICASSP.2014.6854950

Fu, Z., G. Lu, K.M. Ting and D. Zhang, 2011. A survey of audio-based music classification and annotation. IEEE Trans. Multimedia, 13: 303-319.

He, K., X. Zhang, S. Ren and J. Sun, 2016. Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Jun. 27-30, IEEE Xplore Press, Las Vegas, NV, USA, pp: 770-778. DOI: 10.1109/CVPR.2016.90

Hinton, G.E., N. Srivastava, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, 2012. Improving neural networks by preventing co-adaptation of feature detectors.

Howard, A.G., M. Zhu, B. Chen, D. Kalenichenko and W. Wang *et al*., 2017. Mobile nets: Efficient convolutional neural networks for mobile vision applications.

Hu, J., L. Shen and G. Sun, 2017. Squeeze-and-excitation networks.

Ioffe, S. and C. Szegedy, 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift.

Kim, T., J. Lee and J. Nam, 2018. Sample-level CNN architectures for music auto-tagging using raw waveforms. Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Apr. 15-20, IEEE Xplore Press, Calgary, AB, Canada, pp: 366-367. DOI: 10.1109/ICASSP.2018.8462046

Lee, J., J. Park, K.L. Kim and J. Nam, 2017. Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms. Proceedings of the 2nd Music Computing Conference, Jul, 5-7, Espoo, Finland, pp: 220-226.

Liu, J.Y. and Y.H. Yang, 2016. Event localization in music auto-tagging. Proceedings of the 24th ACM International Conference on Multimedia, Amsterdam, Oct. 15-19, Netherlands, pp: 1048-1057. DOI: 10.1145/2964284.2964292

Pons, J., O. Slizovskaia, R. Gong, E. G´omez and X. Serra, 2017. Timbre analysis of music audio signals with convolutional neural networks. Proceedings of the 25th European Signal Processing Conference, Aug. 5-7, Kos Island, Greece, pp: 2744-2748. DOI: 10.23919/EUSIPCO.2017.8081710