Music Genre Recognition

Audio Technology - August 2016 - Lecturer: Günther Schatter

Benedikt S. Vogler
Matrikelnummer 113924
Bauhaus-Universität Weimar
Fakultät Medien - B.Sc. Medieninformatik
benedikt.vogler@uni-weimar.de

Amir Othman
Matrikelnummer 113703
Bauhaus-Universität Weimar
Fakultät Medien - M.Sc. Medieninformatik
amir.othman@uni-weimar.de

ABSTRACT

This writing explains an approach of solving the problem of music genre classification by applying machine learning techniques. We utilized multi-layered artificial neural networks and managed to an accuracy of 83%, which is reasonable for practical purposes.

Keywords

music information retrieval, music genre classification, deep learning, neural networks

1. INTRODUCTION

Today in the 21st century the usage of audio music files have grown enormously. With large amounts of files comes the need to classify the files to organize them. This task is typically done with a lot of manual human labor. Automatic music genre recognition (MGR) is a sub field of music information retrieval (MIR). Algorithms can use features of the sound files found in the sound waves to classify the files.

We want to build our own solution to implement such an algorithm.

1.1 Classification Problem

The question of which genre a music (file) belongs to, is a question of classification - a semantic problem. Music can be classified by its time of formation, geographical origin, topic or a set of rules regarding the sound. Some of these facts are often added to the files as meta data because they can not be retrieved from the sound waves. Humans however classify music by the perception of the sound produced by the audio signal. Music genre is subjective from person to person and can be ambiguous. On top of that, a music file can be assigned more then one genre and using more than one classification category, e.g. ”British music” and ”rock music” can come together as ”Brit-rock”. It is therefore questionable to speak in terms of ”accuracy”, ”hit” or ”miss” if a song can not be objectively assigned a genre. Every accuracy value can therefore only be regarded as some fuzzy approximation.

Music genre classification can either be done by learning the characteristics of collections of songs which their genres are already determined. This is referred to as supervised learning. Another approach is unsupervised learning. In this approach, unlabeled songs are analyzed. By examining their characteristics, the algorithm will then attempt to build clusters of songs based on a similarities.

2. RELATED WORK

A big portion of the people in the field of MGR use artificial neural networks for music genre classification ([1], [9]). Feature choice is an important part of genre classification. MGR benefited a lot of the progress in the field of speech recognition as both rely on feature analysis.

There are also methods where first the chords or the melody are retrieved and then this data is used for machine learning [8]. It is not looked directly at the sound but more at the composition.

Jiang and his team [2] got good results with the usage of the octave-based spectral contrast (OSC) as a feature which they suggested. This was later extended to the octave-based modulation spectral contrast (OMSC) by C.-H. Lee and his team [4]. The OSC is explained further later in this paper.

The choice of the model of the machine learning is also object of examination by researchers e.g. deep neural networks ([3], [10]). Convolutional show good results in text and image analysis and recently are also investigated in MIR [5].

3. THEORETICAL BACKGROUND

Looking at the theoretical background the basic situation is the following: We have two sets and an unknown relation between the sets. The first set is the set of music files, and the other set is the set of genres. To find the relation the use of machine learning is reasonable.

4. DATASET

Our data set must containd two sets (songs and their affiliated genres) which will be used to find the yet unknown relation via machine learning. Many researchers use the GTZAN data set[1]; however, it has been criticized for having flaws [11]. Therefore we compiled our own custom data

[1]A data set consisting of 1000 files at 30 seconds length each, using ten genres, http://marsyas.info/downloads/datasets.html

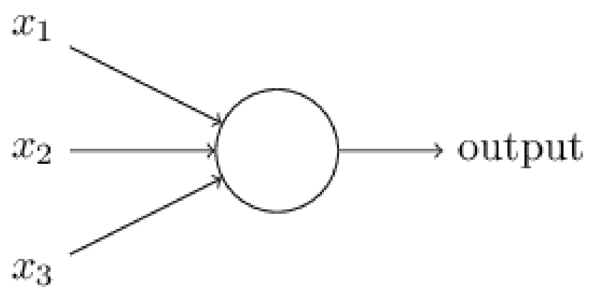

Figure 1: Single Neuron

set as many other researches do so. We used music compilations sold by commercial labels with 100 songs of each genre. Each data set is then split into a training and a test data set. At the end of our work, we saw that there were some files which were not fitting into the classification.

4.1 Neural Network

As explained earlier, the task at hand is a classification task. A classification task can be formulated as a supervised machine learning task. Machine learning provides the ability for a computer program to learn a pattern without explicitly programming the classification rules. Supervised machine learning refers to a subarea of machine learning where the algorithm is provided with labeled data. The algorithm then learns from the labeled data in order to be able to induce the label of previously unseen data. Machine learning typically involves the steps of feature extraction, feature selection, modelling and model evaluation.

Within the field of machine learning, there are different types of models, each with their strengths and drawbacks. We have chosen to use a neural network approach because this approach provides a lot of flexibility in terms of tuning the model and also this approach is currently of great interest within the machine learning community.

A neural network consists of simple computing units known as neurons. A neuron accepts multiple inputs and applies an activation function. The neuron will then fire an output according to this activation function. Each input has its corresponding weights. Intuitively, the weights can be seen as the connection strength between the neurons.

An example of an activation function is a sigmoid function. The activation function determines the output of the neuron. This output will then be propagated to the next connecting neuron or if it is within the output layer it will be the resulting output. Equation 1 is the mathematical form of a Sigmoid function.

$$S(t) = \frac{1}{1+e^{-t}} \tag{1}$$

A network of neurons is formed by interconnecting neurons in layers. The outermost layer is known as the output layer, the innermost layer is knows as the input layer. The layers in between are known as the hidden layer. Neural networks with more than two hidden layer are called deep neural networks. The interconnection of these neurons provide a distributed representation of knowledge throughout the network.

A neural network can be seen as a universal function estimator. We can employ learning algorithms in order to train the neural network. In order for a neural network to fit a specific function, the network will adjust the weights of the neurons according to the learning algorithm. The learning algorithm decides how strong or weak the weights will be.

Typically a learning algorithm operates by propagating the input value from the input layer to the output layer. Then, the values obtained from output layers are then compared with the true value (label value). The difference between the true value and the output value are usually referred to as the error. This value is then propagated backwards from the output to the input. By using optimization techniques, for example gradient descent, the weights can be adjusted in order to minimize the error. The step of propagating the input value to the output layer is known as forward propagation, while the step of propagating the error backwards is known as back propagation.

Neural networks are powerful, but they posses drawbacks regarding speed and efficiency. With increasing numbers of neurons and layers, the required computation resource increases exponentially and can make it impractical to use. Fortunately, the power of Graphical Processing Units (GPUs) have increased throughout the years. GPUs are specialized hardware designed specifically to speed-up the heavy calculation in the realm of graphics. The type of calculation done on GPUs are usually in the form of linear algebra. This happens to be the same family of calculations required in neural networks. This means, we can exploit GPUs to speed up our calculation.

Some of the advantages of neural network includes its robustness against noise. In the case of genre classification, this is definitely true. Our data can be seen as noisy due to the fact that definition of genres can be unclear and subjective.

There are different kinds of neural networks which are good for different tasks. The two families of neural network that we have employed are Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN).

4.1.1 Convolutional Neural Network

Convolutional Neural Networks (CNN) are types of neural network that adds a convolutional layer to the neural network architecture.

Convolution refers to the mathematical operation of integrating the overlap of one function on another kernel function. The kernel function, also known as a filter function, is slided throughout the domain of the source function producing an output which intuitively can be seen as a blend of these two functions. Figure 2 shows an example of a simple convolution operation on a 2D data.

CNN has showed promising results in the real of computer vision and image recognition. The nature of our data is comparable to image data as they are they too form a two dimensional matrix, with each element of the matrix holding a different value. We theorize, that since our data forms patterns that are visually recognizable, algorithms and ideas which are used within image recognition should be able to show favourable results. This argument can be further supported with the fact that the convolutional layer takes advantage of the two dimensional structure of the data.

Another additional component that usually exists in CNNs is the pooling layer. Typically the pooling layer is applied the convolutional layer. Pooling layer subsamples the convo-

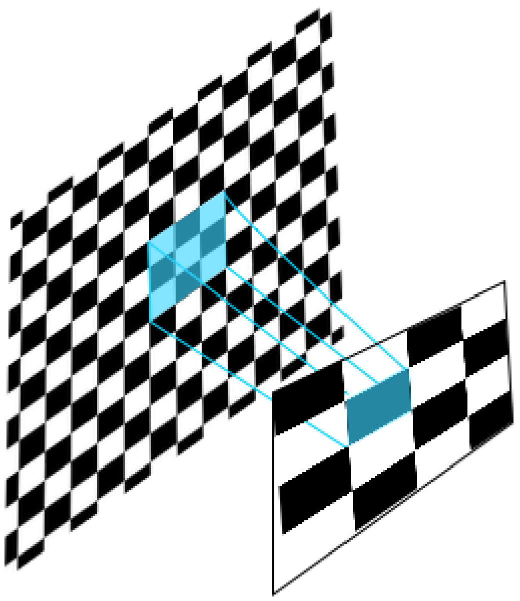

Figure 2: Simple Example of Convolution.

lution produced by the convolutional layer. A pooling layer consist of a filter that holds a function which determines the way of the pooling. One of the more common ways of pooling is applying a maximum operation. The subsampling occurs by the filter selecting the maximum point within the window of the filter.

The MNIST database[2] catalogues the performance of different pattern recognition algorithm on a set handwritten digits. As of this writing (July 2016), the usage of CNNs shows greater results than other techniques for the task of image recognition.

Zhang and Wallace [13] also introduced the usage of CNNs in their study.

4.1.2 Recurrent Neural Network

Another family of neural networks which we hypothesize to be useful to our application are Recurrent Neural Networks. Recurrent neural networks adds loops to the neuron unit. This allows the network to be able to learn sequential dependency.

Audio data is an example of sequential data. Recurrent neural network allows us to model sequential data while still enjoying the flexibility of neural networks. Due to the recurrent connections, training a neural network can be slow.

One of the more common sequential modelling task is the task of language modelling. We therefore took inspiration from this field in order to choose the type of neural network that we should be considering. The type of recurrent neural network that shows the most promising results are the Long-Short Term Memory Recurrent Neural Network (LSTM). The specialty of LSTMs are that instead of just modelling the a time-step by a recurrent connection, it creates two more different types of reccurent connections, namely the long-term and short-term connection. The weights of the connection is determined by the data itself via the learning algorithm.

[2]http://yann.lecun.com/exdb/mnist/

4.2 Features

The choice which features to use is one of huge importance for the accuracy. We chose mel frequency clepstral coefficients (MFCC) and octave-based spectral contrast (OSC).

4.2.1 MFCC

The mel frequency cepstrum coeficcients (MFCC) are the coefficients that make up the mel frequency cepstrum (MFC). The MFC is obtained by several steps. First a signal frame (e.g. 20 ms) is analyzed with fast fourier transformation (FFT) to obtain the spectrum. The logarithm of the spectrum is then taken, put on the mel scala and a discrete cosine transform is applied to obtain the cepstrum[6].

4.2.2 OSC

D.-N. Jiang and his team [2] proposed a novel feature: The octave-based spectral contrast. It is a number which indicates how big the difference between the highest peak and the deepest valley in the spectrum is. First a fast Fourier transformation is performed to obtain the spectrum. The spectrum is then divided into buckets. Usually around eight buckets are used. The buckets are octave sized, so they grow exponentially in size. For every bucket the amplitude is calculated and then the buckets are sorted. After they are sorted in descending order the spectral peak can be calculated with b: the b-th bucket, N_b amount of frequency bins, α: neighbor factor (can control the amount of neighbor buckets); $x_{b,i}$: the spectrum in the b-th bucket, i-th frequency bin

$$Peak_b = \log(\frac{\sum_{i=1}^{\alpha N_b} x_{b,i}}{\alpha N_b}) \tag{2}$$

and the spectral valley with:

$$Valley_b = \log(\frac{\sum_{i=1}^{\alpha N_b} x_{b,N_b-i+1}}{\alpha N_b}) \tag{3}$$

5. TECHNICAL IMPLEMENTATION

First we tested music genre classification with the software 'Neuroph Studio'. This software offers a graphic user interface (GUI). For further work we thought about skipping the GUI and use the Neuroph framework directly. However Neuroph is Java based and designed to only run on the CPU. As machine learning needs high performance computing we need something faster. The framework 'keras'[3] offers massive parallelization on the GPU using the *theano* back-end. It is programmed via the Python programming language.

We wanted to design and implement a pipeline (fig. 4) which allows as input a youtube.com URL and as output gives one of three genres: rock, pop or hip-hop.

For the machine learning we need the feature data. Therefore we have to extract them before we can train the network. Once the network is trained a single file can be put as input and checked with the network. The steps are:

1. Music file download.

2. Feature analysis.

[3]https://github.com/fchollet/keras

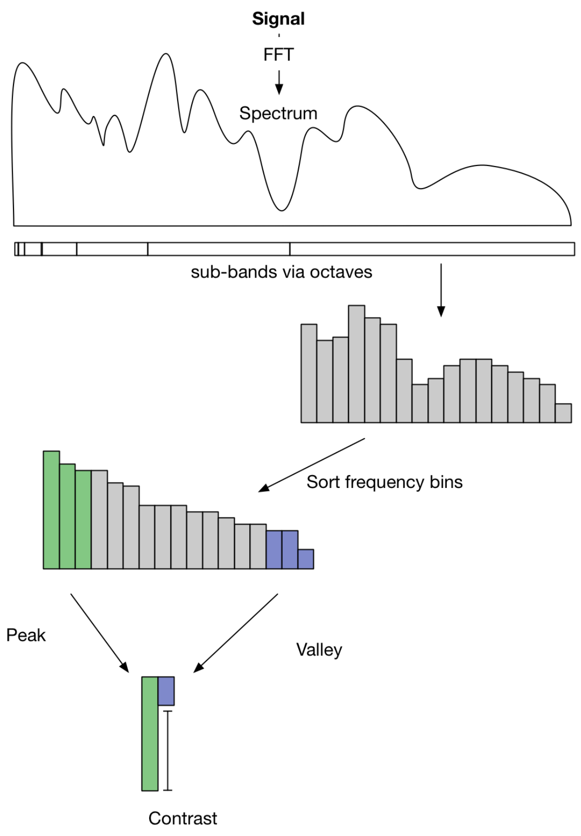

Figure 3: Visual explanaition of the OSC analysis.

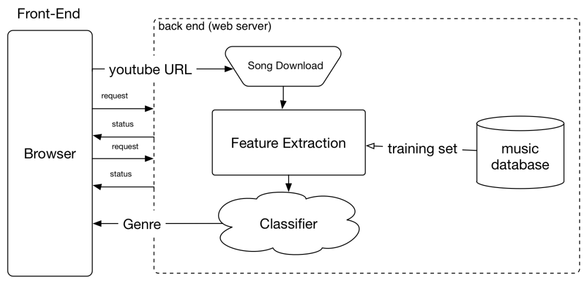

Figure 4: Design of pipeline

3. Features as input for neural network.

For the complete pipeline you need to have the following installed on your system: bash, python3, keras (machine learning framework), hdf5 (file format), h5py (python library for hdf5), youtube-dl (for song download), Vamp plugin system (feature analysis), Vamp BBC plug-ins[4], Vamp QM plug-ins, ffmpeg (30 second split),.

5.1 Feature Extraction

Feature extraction refers to the process of taking audio files and convert them into numerical vectors which describe certain characteristics of the sound waves. The process for using the data for machine learning involves several steps.

We utilized Vamp[5] plug-ins for this purpose. Vamp is an audio processing system for plug-ins that extract descriptive information from audio. In order to use Vamp, we need a host which is suitable for our purpose. Sonic Annotator is command-line tool which is able to host Vamp plug-ins. Sonic Annotator allows us to use Vamp plug-ins in batch and since this is a command-line tool, we can easily run it via Bash scripting for batch processing.

One of the main advantages of using a Vamp plug-ins system is the huge availability of open source implementations of countless plug-ins. We therefore have the freedom to choose and experiment with any audio feature that we need.

Sonic Annotator extracts audio features and outputs them as Comma Separated Values (CSV) files. Each line of a CSV file consists of multiple columns. The first column is the time stamp in seconds. The following columns of the line would be the feature within that time duration.

Sonic Annotator supports batch processes but it will exhibit some difficulty due to the amount of memory used during the extraction. We realized that splitting the files into 30 seconds clips will allow us to avoid the error from happening. This is also in conjunction with our heuristic that humans, when listening to a 30 seconds clip, are able to identify the genre the song belongs to. For splitting the files in 30 seconds we utilize the software ffmpeg.

For extraction we used the BBC for OSC analysis and QM plug-ins for MFCC. For the OSC the documentation stated that only peaks and valleys are extracted and not the contrast. We therefore calculated the difference by our self. However result with peaks and valleys instead of the contrast returned slightly better results

[4]https://github.com/bbcrd/bbc-vamp-plugins/blob/master/README.md
[5]http://vamp-plugins.org

5.2 Vectorization

Once we have the CSV files of audio features, we would like parse and convert these files into vectors which will be able to consumed by the programming environment of our choice. We use Python as our primary programming environment because it offers a huge palette of machine learning and statistical tools which are widely used in research.

The output desired within this step are two vectors, one would be referred as ”Feature Vector” and the other would be referred as ”Label Vector”.

The ”Feature Vector” would be the feature extracted as CSV files rearranged as column vectors.

The ”Label Vector” would be the corresponding label of the feature vector. In our case, the label refers to which genre the song belongs to.

The process of converting CSV files to vectors require a lot processing power and time. Therefore, it is helpful if the vectors are saved locally on the hard disk. For this purpose, we utilize a serialization technique which is implemented in Python. This serialization technique is known as ’pickling’ and it is built in as a part of the standard library of the Python programming language.

Concluding the steps involved in pickling are: audio file $\rightarrow$ csv $\rightarrow$ numpy $\rightarrow$ pickle.

5.3 Modelling

Modelling refers to the crafting of a function estimator that produces the least amount of error. As explained in the previous sections, we utilized different architectures of neural network for this purpose. One of the most convenient and flexible implementation of neural networks that is available for us to use is Keras. Keras is a modular minimalist neural network library implemented in Python. Keras is implemented on top of Theano[12]. Theano is a low-level neural network library which is capable of running on CPU and GPU. Keras can be seen as a high level modular abstraction of Theano. Besides the flexibility of having interchangeable building blocks, the same code can be compiled into code which capable of running on the GPU.

As explained previously, two different neural network architecture were utilized in our implementation. Convolutional Neural Network and Long-Short Term Memory Recurrent Neural Network.

We have built two separate neural networks for the two features that we have chosen. The two neural networks are then concatenated to form a bigger neural network. The reasoning behind building two different neural networks, is so that each neural network can be specialized to a single feature instead of having one bigger neural network which tries to generalize to both features. As we increase the dimension of the feature space, it is harder for the neural network to fit the model.

5.3.1 *Model Evaluation and Hyperparameter Tuning*

One of the drawbacks to neural networks is the huge number of possibilities for the configuration of the architecture of neural networks. We require a systematic method in order to choose the optimum configuration for our neural network.

Before discussing about optimizing the model, we shall discuss in brief how the model is evaluated. The model is evaluated by means of accuracy measure. Accuracy refers to the correct prediction of the model.

Before training the model, the data set is split into training set and testing set. The original data set is split so that the training set makes up 70% of the whole data set and the rest forms the testing set.

The testing set is not used for the training of the model. This means, the label would be unknown to the model while training. This is desired, because if the model uses the testing set while training, it will produce high accuracy but this accuracy is misleading because the model is already fit to the testing set. Only the Testing Set is used to measure the accuracy of the model.

Hyper parameters refers to the higher level parameters of the model which can the behaviour of the model. While regular parameters can be learned by the learning algorithm, hyper parameters are not learned from the learning algorithm. Examples for hyper parameters in our case would be the number of hidden units and the number of filters for our convolution. It is up to the designer to choose the values of these hyper parameters.

One of the approaches to tune hyper parameters is by exhaustively try all the possible configuration of the neural network. This would be reasonable if the number of hyper parameters is small. This is not for our case. Exhaustive search is therefore not practical for our purpose.

Another approach is to use a randomized search. This approach is more reasonable for our purpose.

The way the randomized search is by first choosing a number for a cycle. This number is the number of cycles within which the algorithm will pick randomly the values of hyper parameters. After this cycle, the algorithm analyzes the correlation of the values of the parameters with the accuracy. To measure the correlation we use the value Spearman’s Rank Correlation Coefficient. This value assesses how well the well the relationship between two variables can be described using a monotonic function. The parameter which is the least correlated is penalized by removing the value which produces the lowest accuracy. Then this is repeated until there are no more values for the parameters.

After running the randomized search, we will have the optimum value of the hyper parameters. Then, by using the values of the hyper parameters we train the neural network and save the weights so that we can reuse network to query for genre of an audio file.

To query for a genre, the audio file goes through the whole pipeline which involves the following process:

1. Split the song into sections of 30 seconds length.
2. Extract features from the audio file into CSV files.
3. Parse CSV files into vectors.
4. Predict the label with the trained model.

For each 30 second section, the model predicts a genre. A song of some minutes length results therefore in several genre results. We take the most frequent genre as the genre of the song.

5.4 Front-End Interface

The pipeline is a single executable. It is therefore possible to run a query with one single call which works well together with the Common Gateway Interface (CGI). One call to a web server is needed for starting the query; another call is needed for retrieving the result from the server. The whole

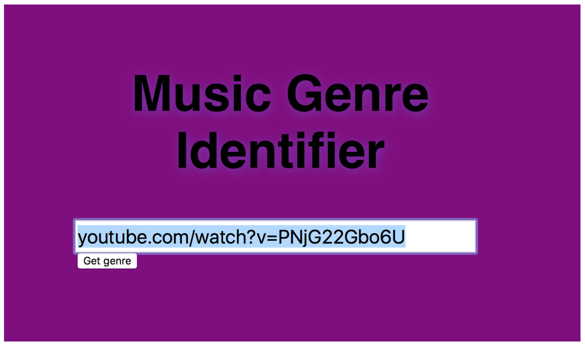

Figure 5: Screenshot of the web interface.

process to analyze the file til the results are ready needs some seconds.

This can be combined with an easy to use web interface. We implemented one (fig. 5) but we had problems with the web server software. For security reasons Apache Server runs the processes with different user rights. This causes problems with the various sub-processes.

The communication between the front-end and the web server can be made via Ajax. A simple and expensive solution is to request the result every x ms until it is available. A more complicated solution is to implement web sockets or long polling to avoid huge amounts of requests. In our implementation a file containing the resulting genre is created after the analysis. Its file name is the youtube-id. It can than be retrieved via HTTP and if not yet finished the web server returns a HTTP 404 error because the file is not there yet. With this design as the web service the computing on the web server can be scaled or modified without involvement of the user or him/she noticing changes. Because of the small range of genres our design is more proof-of-concept then a solid product.

6. RESULTS

Our pipeline works as intended once every component is installed.

6.1 Technical

Sadly we did not manage to get the web server's CGI module properly up and running. The installation process is very complicated because of the high amount of dependencies.

6.2 Detection Results

We achieved an accuracy of 83 %. For comparison, this year's (2016) MIREX results for ten genres were between 70% and 76% [7].

7. FUTURE WORKS

The proposal for the web service software could be implemented. Using more features then two may improve the results. Using different type of neural networks may also return better results.

8. REFERENCES

[1] A. Goel, M. Sheezan, S. Masood, and A. Saleem. Genre classification of songs using neural network. *International Conference on Computer and Communication Technology*, 2014.

[2] D.-N. Jiang, L. Lu, H.-J. Zhang, J.-H. Tao, and L.-H. Cui. Music type classification by spectral contrast features. 2002.

[3] A. Kar and C. Ahuja. Music classification using dnn's. 2014.

[4] C.-H. Lee, J.-L. Shih, K.-M. Yu, and J.-M. Su. Automatic music genre classification using modulation spectral contrast feature. 2007.

[5] T. Lidy. Parallel convolutional neural networks for music genre and mood classification. *MIREX2016*, 2016.

[6] B. Logan. Mel frequency cepstral coefficients for music modeling. 2000.

[7] Mirex. Mirex 2016: Audio train/test: Genre classification (mixed) results, last checked 13th august 2016. http://www.music-ir.org/nema_out/mirex2016/results/act/mixed_report/summary.html, 2016.

[8] A. Nasridinov and Y.-H. Park. A study on music genre recognition and classification techniques. *International Journal of Multimedia and Ubiquitous Engineering*, 9(4):31–42, 2014.

[9] N. Prabhu, A. Asnodkar, and R. Kenkre. Music genre classification using improved artificial neural network with fixed size momentum. *International Journal of Computer Applications*, 101(14), September 2014.

[10] S. Sigtia and S. Dixon. Improved music feature learning with deep neural networks. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6959–6963, 2014.

[11] B. L. Sturm. The gtzan dataset: Its contents, its faults, their effects on evaluation, and its future use. *arXiv:1306.1461v2 [cs.SD] 7 Jun 2013*, 2013.

[12] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016.

[13] Y. Zhang and B. C. Wallace. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv*, April 2016.