Classification of music genre in songs

Fabian Längle
Computer Based Sound Production
Faculty of Computer Science
University of Ljubljana

***Abstract*—The purpose of this seminar work is to classify the music genre of song snippets by training a convolutional neural network (CNN) with time-frequency spectrograms of the respective song. The field of genre classification has only recently emerged into the benefits of fully automated classification algorithms with the help of state-of-the-art neural networks. A transfer-learning approach using the well-known and open source VGG16 deep learning model peaked in a validation accuracy of 59 %. This was achieved on a dataset of 1000 song snippets equally distributed into 10 genres.**

I. Introduction

Not too long ago the classification of songs into genres by considering the rhythm structure and harmonics was a highly manual task [1]. This caused enormous amount of work going along with decreasing creation cycles and therefore the shorter time needed to create new music pieces. Since genres are an extremely helpful tool to organize music files there was a rising demand for automated genre classification. There are many ways to extract information from a song in order to classify its genre, e.g. using the time and frequency domain. A rather new approach uses a time-frequency spectrogram to represent each song and detects the genre by applying a trained CNN on the given songs. This strategy has the potential to compete with the accuracy of established genre recognition algorithms. This seminar work will implement a CNN. The model implementation will be done in Python by using well-known, open source libraries specified on sound analysis and machine learning.
The following seminar work first describes and compares two related works in this field. The used methods of this work and the general approach are described, followed by the results. Lastly the findings are discussed in the conclusion.

II. Related work

Tzanetakis et al. are the initiators and creators of the GTZAN dataset. The purpose of this paper was not only to create a dataset but moreover to extract features and perform first classification models [1]. The features used for classification in this paper (e.g. Timbral Texture Features, Spectral Flux, Mel-Frequency Cepstral Coefficients etc.) are mostly based on a short time Fourier transformation (STFT). Since they mainly have numerical characteristics they can be used for classic machine learning approaches [1]. This differs from the approach chosen in this seminar work. Nevertheless, helpful insights about the dataset will be considered.

In another paper Costa et al. follows a similar approach to the one chosen in this seminar work. It is an ensemble approach which combines the outcomes of three different classifiers. The first extracts acoustic features from the raw audio file and inputs them into a Support Vector Machine (SVM) classifier. The other two classifiers are based on the visual representation of the song (spectrogram): Deriving handmade features and inputting them into another SVM whereas the third classifier is training a CNN with the spectrograms [2]. Therefore one of the ensemble classifiers is similar to the approach followed in this seminar work. Creating a meta algorithm based on parallel ensemble can lead to results that are better than every part of the ensemble could have reached individually. The same effect can be observed in a negative way.

III. Methods

The available GTZAN dataset (1.3 GB) is comprised of 1000 song snippets of 30 seconds each. These song snippets are equally distributed into ten different genre categories. This is the starting point for following steps.

A. Data preprocessing

For loading audio files in python and converting the raw audio into machine readable data *librosa* package is essential. The audio file is read in with a sample rate of 22050 Hz which is sufficient for this seminars purpose. To create input data for the CNN the audio files have to be turned into visual representations. This is done by converting every song snippet into a spectrum of its frequencies over time - a so called spectrogram. A short time Fourier transformation (STFT) is applied to every snippet to extract the magnitude of a given frequency within a time frame. More specifically the y-axis represents a MEL-spectrogram which comes closer to the representation of human perception of music. For the Fourier transformation a window length of 2048 is chosen. Four sample MEL-spectrograms of the first four songs originating in different genres (Disco, HipHop, Metal and Classical) are displayed in figure 1. A human eye can already detect varying patterns between the four graphs. That means there is a realistic chance that a CNN could detect these differences and reproduce the learned rules on unseen data.

B. Initial model

The first step for building a CNN is to make the input images machine readable by turning each image into an array.

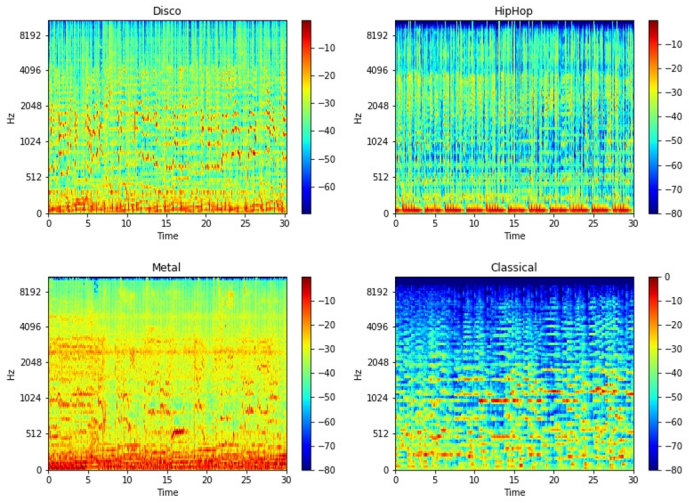

Fig. 1. Spectrograms of four different genres.

Afterwards the correct label to each picture is extracted and one hot encoded to make it compatible for the neural network. The whole data array and the labels are now split into a train and a test set through the built-in train-test-split function from sklearn.

An initial implementation of the CNN has a combination of 8 hidden layers displayed in figure 2. A keras sequential

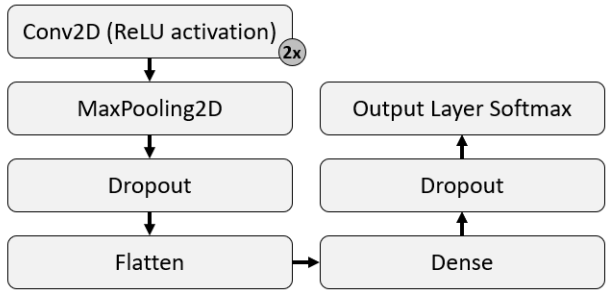

Fig. 2. Layers used to build the initial classification model

model is used to add layer by layer. The first two layers are 2D convolution layers with 64 and 32 filters and a 3x3 kernel each for first feature extractions from the image (e.g. edges). Rectified Linear Activation (ReLU) is the activation function used since it is widely proven in the field of neural networks. The first layer also takes the input shape of the images as an argument. As the spectrograms are colored and therefore have three layers of pixels (RGB) it is sufficient to crop down the images into a 128x128 pixels format. These two layers can be summarized as the first stage of the CNN, the so called *convolutional layer* having the purpose of extracting features from the image.

The following two layers build the second stage, the *pooling layer*. It is reducing the dimensionality while not discarding important features. It is composed of a MaxPooling2D layer with a 2x2 pool size to reduce the amount of parameters and the Dropout layer is added to prevent overfitting on training data by ignoring a given percentage of neurons while training the model.

The remaining layers build up the *fully connected layer*, the third and last stage of the CNN architecture. As part of it the Flatten layer reduces the tensor into one dimension. The Dense layer changes the output size of the layer to 128 with a ReLU activation function followed by another Dropout layer. Finally the last Dense layer reduces the output to 10 according to the number of prediction classes using a softmax activation function. Because softmax makes the values of the output classes sum up to 1 they can be interpreted as probabilities.

C. Transfer learning approach

To improve the model an often chosen strategy is to apply transfer learning. This is a strategy of taking the knowledge of a pre-trained model and using it as a base model for another problem. This way the model already has weights (features) and can detect e.g. edges which safes time while developing a model.

For this seminar work the VGG16 convolutional neural network is used as a base model. It was proposed by K. Simonyan and A. Zisserman and was trained on the ImageNet database containing about 14 million images of 1000 classes [3]. Its architecture is build up of five blocks containing multiple 2D convolutional layers and one MaxPooling2D layer each. This architecture is extended with some additional layers to target the genre classification problem of this seminar work. The on top architecture is displayed in figure 3. The on top

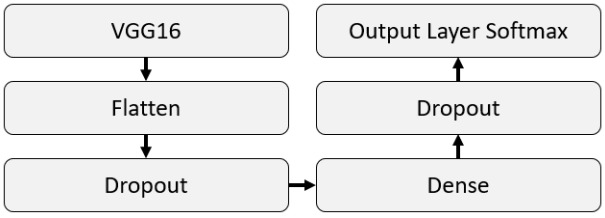

Fig. 3. On top architecture used to build the tranfer classification model

architecture consists of five layers in total starting with a Flatten layer. The first Dropout layer has a dropout rate of 0.2 and is followed by a Dense layer using the L2 regularizer. The L2 value is set to 0.001 to achieve a slow learning rate and prevent possible overfitting. The last Dropout layer has an increased rate of 0.4 before the Output layer applies the softmax activation function to classify the input into the final 10 classes.

Since the VGG16 base model is already trained it can be set to be non-trainable which results in only training the on top architecture during the model fit. This comes with the downside of missing out the parameter fine tuning towards the current application but safes a lot of time during the model training.

IV. Results

The initial CNN model results in an accuracy of about 40 % within 20 epochs and is therefore a rather weak classifier. This is owed to the rather simple architecture of this model. The transfer learning approach led to an accuracy of about 59 % which is a gratifying improvement compared to the initial CNN model. Since the transfer model outperforms the initial CNN model it will be the main focus for the rest of this seminar work.

Figure 4 shows the course of train and test accuracy with a steady improvement in the train curve over the epochs. Test accuracy fluctuates around 50 % and peaks in the third epoch.

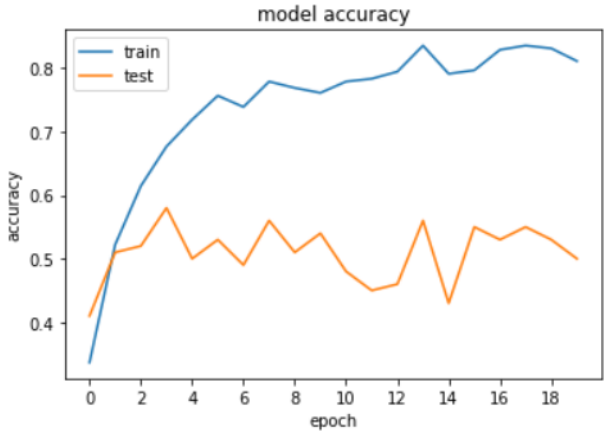

Fig. 4. Development of model accuracy for test and train set

The curve of the model loss shows a less typical shape. The train loss drops constantly over the epochs as expected whereas the test loss continues to increase with further training. This results in a peaking test loss in the 20th and last epoch shown in figure 5.

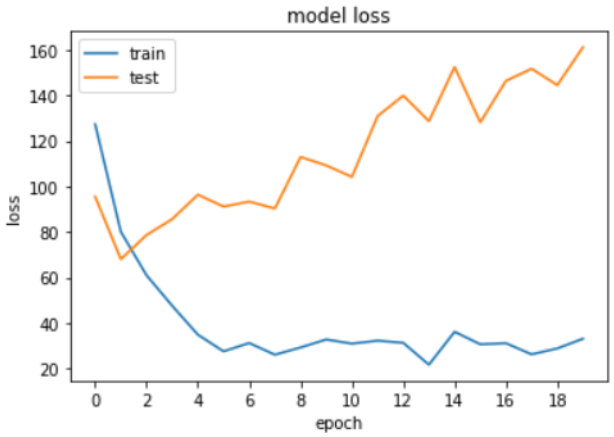

Fig. 5. Development of model loss for test and train set

The confusion matrix of the model (see figure 6) shows a clear diagonal line from the upper left corner down to the lower right corner representing the true positives and true negatives and is therefore indicating correctly classified spectrograms. Nevertheless, some wrongly classified outliers can be observed apart from the diagonal. Especially the class HipHop was weakly classified since not a single spectrogram was predicted correctly into this class.

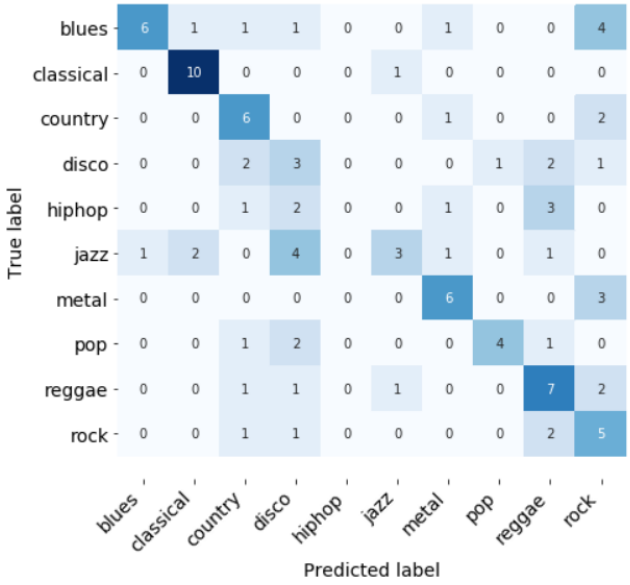

Fig. 6. Confusion matrix of the transfer model

V. Conclusion

The transfer model achieving a precision of 59 % is a good starting point for a genre classifier built with the computational limitations given in this seminar work. The functionality of this model as an operative classifier in a more professional context is not recommended. A more thorough tuning of hyper-parameters can be done to accomplish a boost of precision.

Furthermore, the development of the test loss curve can be alarming. The steady rising trend of the test loss combined with the asymptotic train loss curve towards 0 continues throughout all the 20 epochs and could be interpreted as an potential indicator for overfitting. Introducing weight regularization in form of L2 as well as Dropout layers are well known strategies to fight overfitting and have been applied to the model. Next steps can be to lower the learning rate even further or to increase the dropout rate. Lastly a augmentation techniques (e.g. scaling, rotating or flipping) can be applied to the underlying dataset and therefore artificially increase size of the dataset.

References

[1] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on speech and audio processing*, vol. 10, no. 5, pp. 293–302, 2002.

[2] Y. M. Costa, L. S. Oliveira, and C. N. Silla Jr, "An evaluation of convolutional neural networks for music classification using spectrograms," *Applied soft computing*, vol. 52, pp. 28–38, 2017.

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.