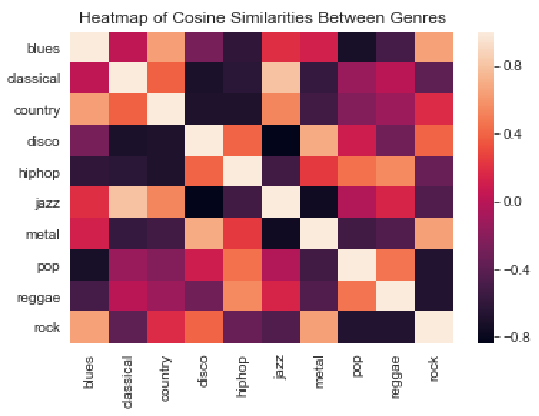

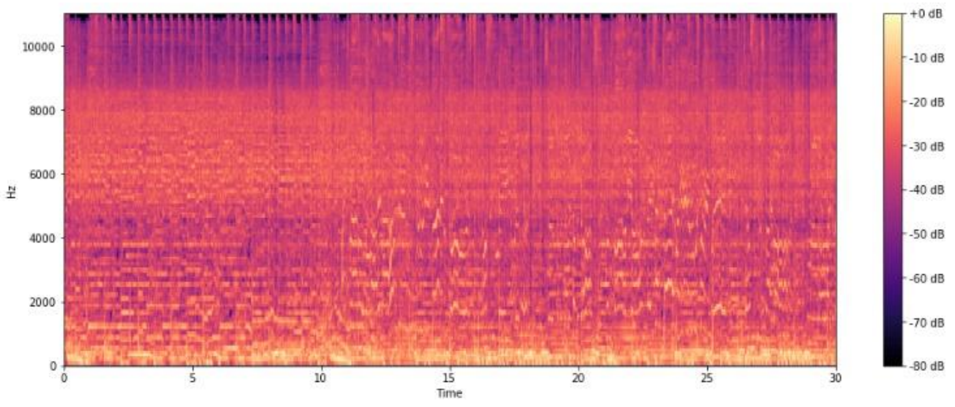

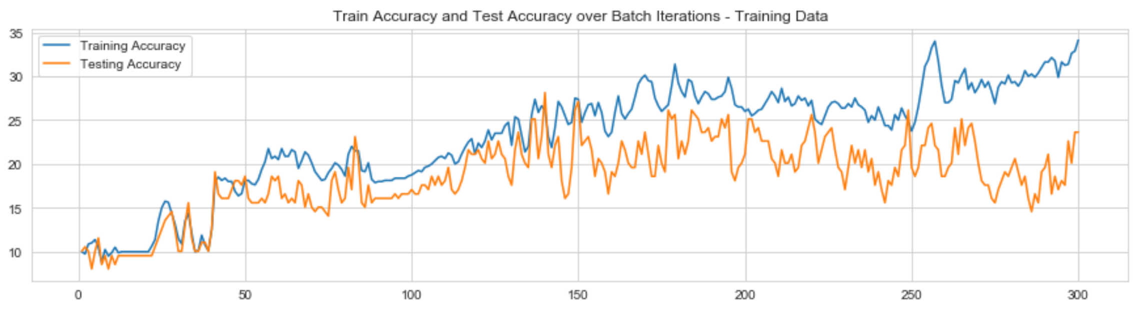

Musical Genre Classification

DATA ENGINEERING, FEATURE EXTRACTION AND CLASSIFICATION OF AUDIO

Rory Cox | 12.03.2020

Project Outline

To design a system to classify audio files into musical genres. The dataset provided was a set of 1000 audio files of approximately 30 seconds, split into 10 genres of 100 files. A client intends to use these files to build a Convolutional Neural Network (CNN) to classify audio into the 10 specified musical genres, and to train a separate classification model using features of the audio files. This project's task was to process the audio files such that they can be classified into genres using a convolutional neural network, and also to extract features from the audio files, and prepare them for use in the second machine learning classification model. Finally, there was the option to perform both classification tasks and report upon their results.

INITIAL CHECKS

After retrieving the audio files, checks were performed to ensure the files were distributed equally among genres, and the number of files were as expected. Since the number of files was small, a visual inspection of the file extensions was sufficient to ensure that the file types were the same. While each file in this dataset was nearly exactly 30 seconds, a measure was implemented to crop files that were longer than 30 seconds in duration, and pad files with 0 values that were shorter than 30 seconds. This is significant when later converting these files to images. For example, in an extreme case where an audio file is 15 seconds in duration, saving this file to an image that is the same size as files that are 30 seconds in duration would drastically stretch the image, and result in a convolutional network's filters being unable to identify features that it would have learned to extract from audio files of the standard duration. Padding solves this since it retains the original scale, as does cropping for larger files, though at the expense of some of the data.

FEATURE EXTRACTION

The Librosa package was used to extract popular features that are commonly used to describe and classify audio files. These features are listed in the project notebook, and are described in detail in the following link. For each audio file, these features were extracted and inputted into a csv along with the file name and label, in a tidy format compatible with supervised learning models.

AUDIO TO IMAGE CONVERSION AND PROCESSING

Convolutional Neural Networks are most commonly applied to image data, since they classify using filters that iterate over data, extracting features based on proximity. Therefore, the audio files were converted to an image format for use in a Convolutional Neural Network supervised learning task. Librosa functions were used to convert image files to Mel-Spectograms. Mel-Spectograms are a much more detailed representation of audio than a simple waveform, displaying frequency and volume over time. It also better represents the human audio range (which is the original basis of the labelling of the audio files), as the Mel scale uses a form of logarithmic spacing which resembles the resolution of the human auditory system. Each Mel-Spectogram image, with the default scale and colour bar removed, is uploaded to folders, and split into 80% and 20% between train and test folders respectively. The DPI and figure size are set to ensure consistency across different devices.

EXPLORATORY DATA ANALYSIS OF THE FEATURES DATASET

Initial inspection of the features dataset confirmed a tidy format compatible with supervised learning models. There were no missing values, and each variable was loaded as the correct data type. Summary statistics were viewed to provide an overview of each variable. To gain an understanding of how each feature corresponds to each genre, Boxplots of each variable by genre were created.

Genres of music are an entirely human construction with no objective mathematical basis for classification. As such, it was important to gain an understanding of the similarities between genres to explain any potential issues that may arise in model performance. Cosine similarity is a measure of the similarity of 2 vectors, in this case, the extracted features for each track. To demonstrate cosine similarity, a function was created to compute the cosine similarity of an input (a track name), and return the 5 most similar tracks. The cosine similarity proved very effective, for example, when a Beastie Boys track was inputted, the 5 most similar tracks contained 3 other audio tracks also by the Beastie Boys. Following this, a matrix of cosine similarities between genres (using the mean value of each feature, grouped by genre) was computed. Classical and jazz were observed to have high similarity, while disco and classical had high negative similarity; as such, it might be expected that models may experience difficulty in differentiating between classical and jazz using the extracted features.

Further EDA was conducted to check for data abnormalities. Histograms can reveal issues that may arise from irregularities in features. For example, In the California Housing Dataset, the variable "house price" is capped at 500,000 USD, such that houses valued over this number are brought down to this value. The result is that any model using house price as an explanatory or response variable uses unreliable data for houses valued over 500,000 USD. A histogram of each feature would reveal this irregularity as an unusually large bar at 500,000 USD. The histograms of the features extracted from the audio revealed no such irregularities, with most distributions appearing close to normally distributed. If linear models were to have been used for classification, techniques such as log transformations might have been necessitated to counteract the skew in some distributions. Multicollinearity was then investigated and addressed. Explanatory variables that are strongly linearly correlated serve only to increase the feature space and introduce unnecessary noise. One variable can be retained and the others safely discarded. Using a heatmap revealed that the spectral centroid variable was strongly linearly correlated with spectral roll-off and zero crossing rate. It was also strongly correlated with the second Mel-frequency cepstral coefficient, though not linearly. Accordingly, spectral roll-off and zero crossing rate were removed from the feature space.

Outlier detection using boxplots of each variable revealed a number of extreme values, and inspecting them individually revealed a corrupted file (reggae86) which was removed from the dataset and images folder. Other outliers such as one that was very vocally-centric for its genre (disco47), may have been unusual in the dataset, but there was no objective basis upon which to remove them.

CONVOLUTIONAL NEURAL NETWORK (CNN)

A convolutional neural network was then designed to classify the Mel-Spectograms of each audio file. Pytorch was used for this task. The CNN design used dropout on linear layers to counteract the effects of overfitting, ReLU activation functions on linear layers to counteract the vanishing gradient problem, a learning rate of 0.002, an Adam optimizer which is commonly used for images, a batch size of 32 to speed training and 300 epochs. A testing accuracy of 23% was achieved, which is quite poor given that humans are understood to be able to classify audio into musical genres with approximately 70% accuracy. Loss and accuracy graphs for both the training and testing data were generated, as well as a confusion matrix, which reveals that the model failed to classify any tracks as disco or hiphop. The results were saved using the pickle package in order to avoid having to retain the model, which can be computationally expensive. The model state was also saved and it was then demonstrated that the model could be loaded and individual image files ran through the network to output a genre. A graphical representation was included to show a probabilistic output of the model, since the model uses a SoftMax activation function.

FEED FORWARD NEURAL NETWORK ON FEATURE SET

Using Keras, after scaling the feature set from the csv, the data was split into train and test sets of 80% and 20%, respectively. Using ReLU functions, an Adam optimizer and the SoftMax activation function, a Feed Forward Neural Network was trained for 20 epochs with a batch size of 128. Aside from the slight overfitting, the model performed exceptionally well on the features database, achieving a test accuracy of 88%, and a train accuracy of 98%. The model state was saved, loaded, and used to demonstrate the ability to classify individual audio files.

CONCLUSIONS

While the task requirements were satisfied, a number of different approaches could also have been taken which may have affected the quality of the results. These include extracting different features for use in the features csv, using principle component analysis to identify uncorrelated features for use in classification, or using more standard classification models such as random forest or linear models before choosing an optimal model using grid-search. Using the features for classification proved much more successful than using the Mel-Spectograms, both in terms of accuracy and time. This leads to the conclusion that either a much improved CNN model architecture is required to classify the images, or that the features obtained using computer vision on a Mel-Spectogram are not sufficient to adequately predict the genre of an audio file, and that musical genres are better described by mathematical features such as measures of signal energy. The dataset should also be discussed. A 30 second snippet of audio is often unrepresentative of the full audio track, which affects the reliability of the label. It might also be of interest to further split the audio data into 3 second snippets, since this is approximately the time that it takes for humans to recognize and classify an audio file's genre, and provide the models with larger datasets.