Matan Lachmish

Prof. Lior Wolf

TAU - Deep Learning

1 May 2016

Music Genre Classification Using Deep Learning

Matan Lachmish
mlachmish@gmail.com

1 Abstract

Throughout the course we mainly focused around computer vision tasks and a little bit of NLP. I decided to reach out a new field and use the deep learning techniques I learned on the field of sound processing.
This paper discuss the task of classifying the music genre of a sound sample.

2 Introduction

When I decided to work on the field of sound processing I thought that genre classification is the parallel problem to the image classification we had learned in class. To my surprise I did not found to many works in deep learning that tackled this exact problem.

One paper that did tackle this classification problem is Tao Feng's paper [1] from the university of Illinois. I did learned a lot from this paper, but honestly, they didn't

reached impressive results (as Ill explain later).

So I had to look on other, related but not exact papers.

A very influential paper was Deep content-based music recommendation

[2] This paper is about content-base music recommendation using deep learning techniques. The way they got the dataset, and the preprocessing they had done to the sound had really enlightened my implementation.

Also, this paper was mentioned lately on “Spotify” blog [3]. Spotify recruited a deep learning intern that based on the above work implemented a music recommendation engine. His simple yet very efficient network made me think that Tao’s RBM was not the best approach and there for my implementation included a CNN instead like in the Spotify blog.

One very important note is that Tao’s work published result only for 2,3 and 4 classes classification. Obviously he got really good result for 2 classes classification, but the more classes he tried to classify the poorer the result he got. My work classify the whole 10 classes challenge, a much more difficult task.

A sub task for this project was to learn a new SDK for deep learning, I have been waiting for an opportunity to learn Google’s new TensorFlow[4]. This project is implemented in Python and the Machine Learning part is using TensorFlow.

3 The Dataset

Getting the dataset might be the most time consuming part of this work. Working with music is a big pain, every file is couple of MBs, there are variety of qualities and parameters of recording (Number of frequencies, Bits per second, etc…). But the worst issue is **copyright**, there are no legit famous songs dataset as they would cost money.

Tao's paper based on a dataset called GTZAN[5]. This dataset is quit small (100 songs per genre X 10 genres = overall 1,000 songs), and the copyright permission is questionable. This is from my perspective one of the reasons that held him from getting better results.

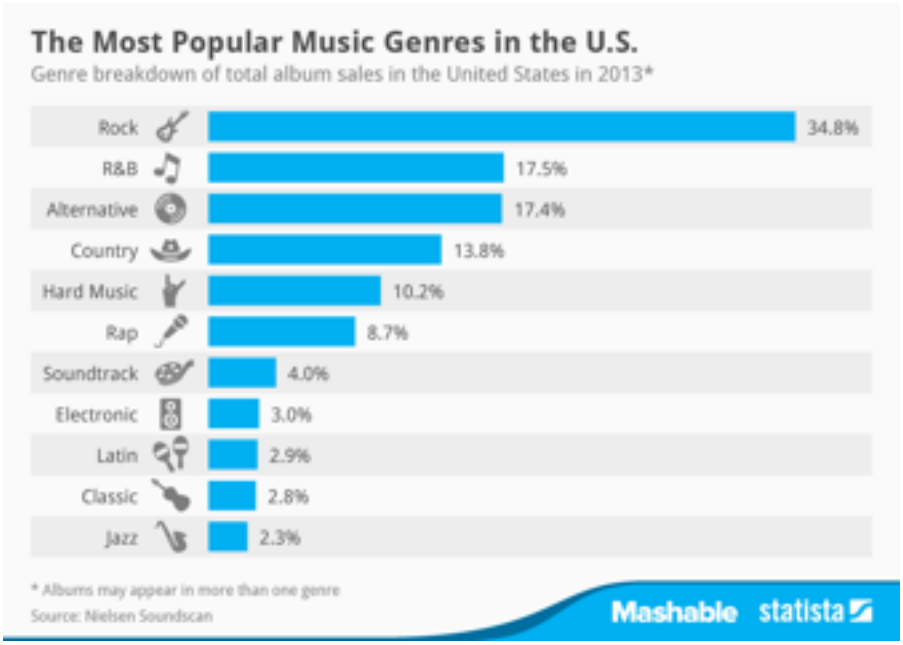
So, I looked up for generating more data to learn from. Eventually I found MSD[6] dataset (Million Song Dataset).
It is a freely-available collection of audio features and metadata for a million contemporary popular music tracks. Around 280 GB of pure metadata. There is a project on top of MSD called tagtraum[7] which classify MSD songs into genres.

The problem now was to get the sound itself, here is where I got a little creative. I found that one of the tags every song have in the dataset is an id from a provider called 7Digital[8]. 7Digital is a SaaS provider for music application, it basically let you stream music for money. I signed up to 7Digital as a developer and after their approval i could access their API. Still any song stream costs money, But I found out that they are enabling to preview random 30 seconds of a song to the user before paying for them. This is more than enough for my deep learning task, So I wrote "previewDownloader.py" that downloads for every song in the MSD dataset a 30 sec preview.

Unfortunately I had only my laptop for this mission, so I had to settle with only 1% of the dataset (around 2.8GB).

The genres I am classifying are:

1. blues
2. classical
3. country
4. disco
5. hiphop
6. jazz
7. metal
8. pop
9. reggae
10. rock



WWW.STATISTA.COM - THE MOST POPULAR MUSIC GENRES IN THE U.S.

4 Preprocessing the data

Having a big data set isn't enough, in oppose to image tasks I cannot work straight on the raw sound sample, a quick calculation: 30 seconds × 22050 sample/second = 661500 length of vector, which would be heavy load for a convention machine learning method.

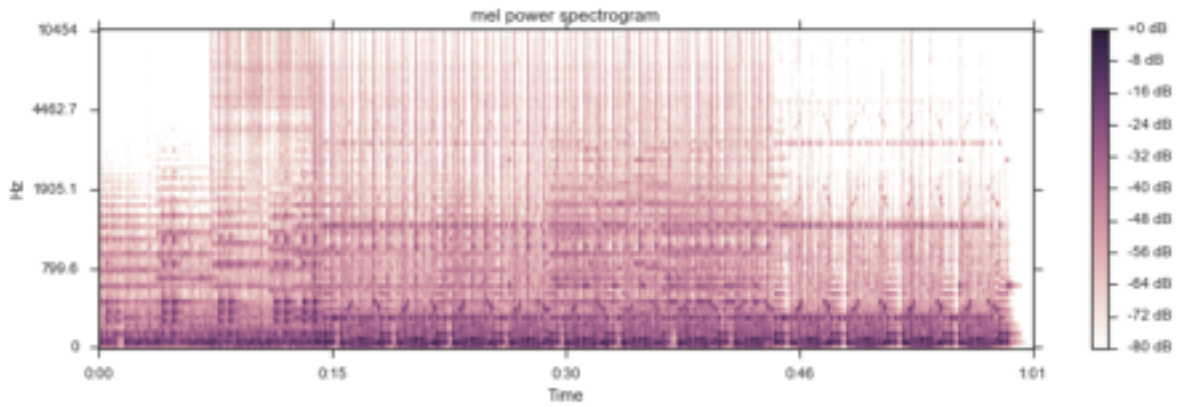
Following all the papers I read and researching a little on acoustic analysis, It is quit obvious that the industry is using **Mel-frequency cepstral coefficients** (MFCC) as the feature vector for the sound sample, I used librosa[9] implementation.

MFCCs are derived as follows:

1. Take the Fourier transform of (a windowed excerpt of) a signal.
2. Map the powers of the spectrum obtained above onto the mel scale, using triangular overlapping windows.
3. Take the logs of the powers at each of the mel frequencies.
4. Take the discrete cosine transform of the list of mel log powers, as if it were a signal.
5. The MFCCs are the amplitudes of the resulting spectrum.

I had tried several window size and stride values, the best result I got was for size of 100ms and a stride of 40ms.

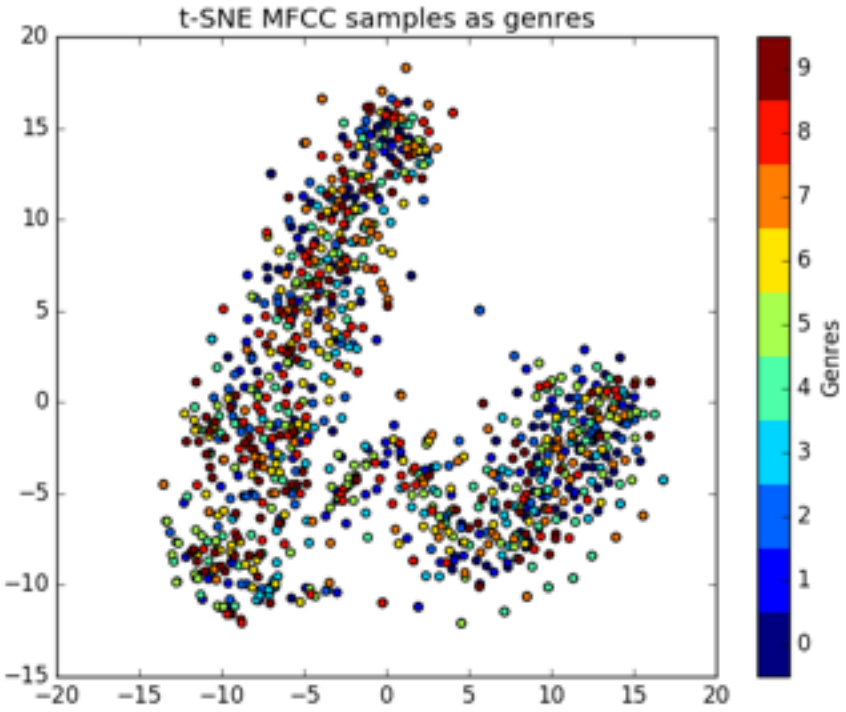
One more point was that Tao's paper used MFCC features (step 5) while Sander used strait mel-frequencies (step 2).

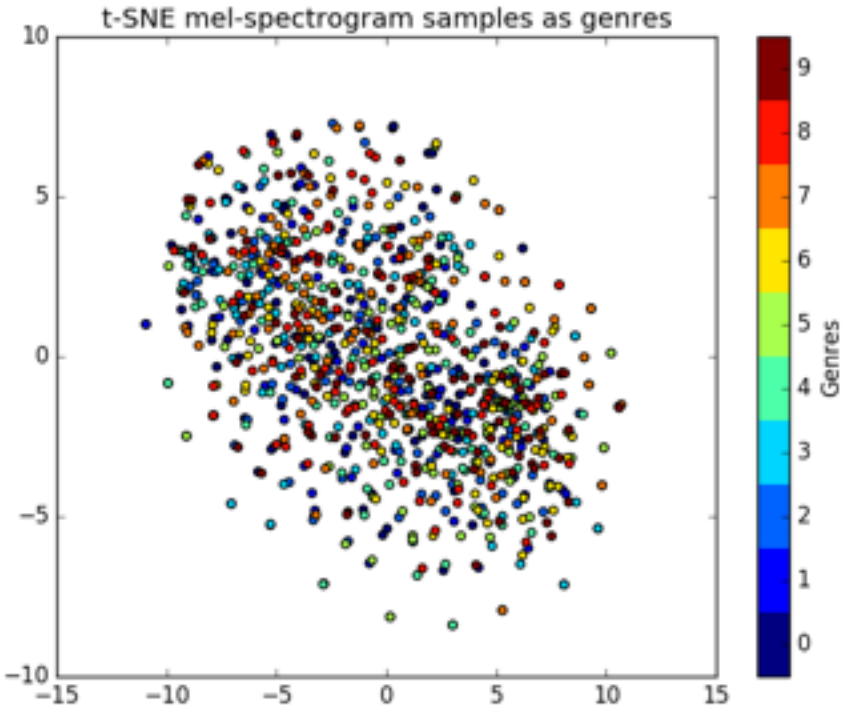


MEL POWER OVER TIME

I tried both approaches and found out that I got extremely better results using just the mel-frequencies, but the trade-off was the training time of-course.

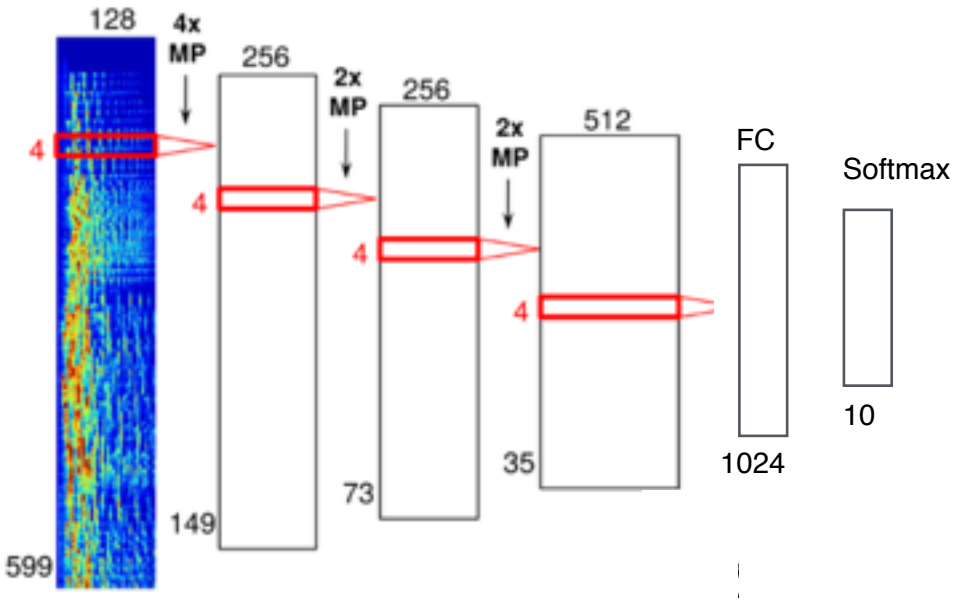
Before continue to building a network I wanted to visualise the preprocessed data set, I implemented this through the t-SNE[10] algorithm.Below you can see the t-SNE graph for MFCC (step 5) and Mel-Frequencies (step 2):





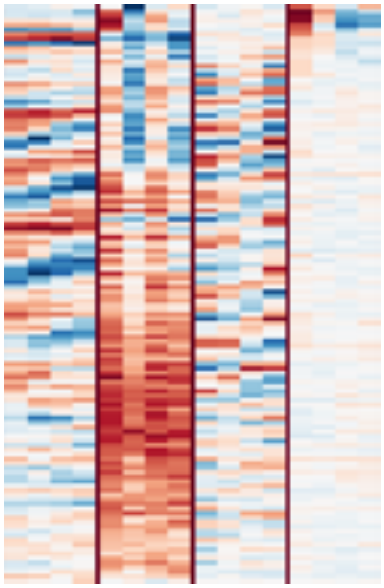
5 The Graph

After seeing the results Tao and Sander reached I decided to go with a convolutional neural network implementation. The network receive a 599 vector of mea-frequency beans, each containing 128 frequencies which describe their window. The network consist with 3 hidden layers and between them I am doing a max pooling. Finally a fully connected layer and than softmax to end up with a 10 dimensional vector for our ten genre classes



I did implement another network for MFCC feature instead of mel-frequencies, the only differences are in the sizes (13 frequencies per window instead of 128).

Visualisation of various filters (from Sander's paper):



- Filter 14 seems to pick up **vibrato singing**.
- Filter 242 picks up some kind of **ringing ambience**.
- Filter 250 picks up **vocal thirds**, i.e. multiple singers singing the same thing, but the notes are a major third (4 semitones) apart.
- Filter 253 picks up various types of **bass drum sounds**.

6 Results

As I explained in the introduction, the papers I based my work on did not solve the exact problem I did, for example Tao's paper published results for classifying 2,3 and 4 classes (Genres).

Tao Feng	
4-genre - DBN	51.88
4-genre - NN	63.75

I did looked for benchmarks outside the deep learning field and I found a paper titled "A BENCHMARK DATASET FOR AUDIO CLASSIFICATION AND CLUSTERING" [11]. This paper benchmark a very similar task to mine, the genres it classifies: Blues, Electronic, Jazz, Pop, HipHop, Rock, Folk, Alternative, Funk.

A BENCHMARK DATASET FOR AUDIO CLASSIFICATION AND CLUSTERING	
Random	26.72
C4.5	45.44
Naive Bayes	43.69
k-NN	53.23
CNN with mel spectrum (My results)	30.02

My results:

My results	
CNN with mel spectrum 10 classes	46.87
CNN with MFCC 10 classes	17.18

6 Code

- previewDownloader.py:

USAGE: python previewDownloader.py [path to MSD data]

This script iterate over all '.h5' in a directory and download a 30 seconds sample from 7digital.

- preproccess.py:

USAGE: python preproccess.py [path to MSD mp3 data]

This script pre-processing the sound files. Calculating MFCC for a sliding window and saving the result in a '.pp' file.

- formatInput.py:

USAGE: python formatInput.py [path to MSD pp data]

The script iterates over all '.pp' files and generates 'data' and 'labels' that will be used as an input to the NN.

Moreover, the script output a t-SNE graph at the end.

- train.py:

USAGE: python train.py

This script builds the neural network and feeds it with 'data' and 'labels'.

When it is done it will save 'model.final'.

8 References

[1] Tao Feng, Deep learning for music genre classification, University of Illinois. https://courses.engr.illinois.edu/ece544na/fa2014/Tao_Feng.pdf

[2] Aa¨ron van den Oord, Sander Dieleman, Benjamin Schrauwen, Deep content-based music recommendation. http://papers.nips.cc/paper/5004-deep-content-based-music-recommendation.pdf

[3] SANDER DIELEMAN, RECOMMENDING MUSIC ON SPOTIFY WITH DEEP LEARNING, AUGUST 05, 2014. http://benanne.github.io/2014/08/05/spotify-cnns.html

[4] https://www.tensorflow.org

[5] GTZAN Genre Collection. http://marsyasweb.appspot.com/download/data_sets/

[6] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The Million Song Dataset. In Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011), 2011. http://labrosa.ee.columbia.edu/millionsong/

[7] Hendrik Schreiber. Improving genre annotations for the million song dataset. In Proceedings of the 16th International Conference on Music Information Retrieval (ISMIR), pages 241-247, 2015.

http://www.tagtraum.com/msd_genre_datasets.html

[8] https://www.7digital.com

[9] https://github.com/bmcfee/librosa

[10] http://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html

[11] Helge Homburg, Ingo Mierswa, Bu¨lent Mo¨ller, Katharina Morik and Michael Wurst, A BENCHMARK DATASET FOR AUDIO CLASSIFICATION AND CLUSTERING, University of Dortmund, AI Unit. http://sfb876.tu-dortmund.de/PublicPublicationFiles/homburg_etal_2005a.pdf