Deep Neural Networks for Piano Music Transcription

Diego González Morín

Abstract

In this paper the main approaches of Automatic Music Transcription using Neural Networks are reviewed and explained. As an experimental study to these approaches, different models of Neural Networks are proposed to be compared for the application of Polyphonic Piano Music Transcription. This experimentation is first focused on the dataset preprocessing and alignment and is continued by a empirical comparison between Deep Neural Networks and Long Short-Term Memory (LSTM) Networks performances. The objective of the current project is to serve as a first step for future Neural Network design and optimization for Automatic Music Transcription by enclosing the best combination of methods and parameters for this particular task.

1 Introduction

Automatic music transcription is considered a significant step in the field of music signal processing. Some of the applications of Automatic Music Transcription(AMT) are : music information retrieval, music processing (such as changing the instrument set), human-computer interaction [7] or even some commercial Music Transcriptors. Many researchers have worked in different methods to improve the performance of automatic music transcriptors throughout the last decades. However, few actual improvements were achieved during the recent years. Low computational power and obsolete algorithms impose a ceiling for AMT accuracy. The parallel improvement and development of Deep Neural Networks and computational power restored the interest in Automatic Music Transcription, and incited the arise of investigations focused on this new approach.

The main goal of this project is, after reviewing the different techniques and methods for data preprocessing and AMT implementations, experiment with different data preprocessing methods (Mel Filterbank Cepstrum Coefficients, Constant Q Transform) and a small variety of Neural Networks (Dense Neural Networks and Long Short Time Memory Networks). The model architecture, parameter settings, data preprocessing and representation will be described in the Section 2. The proceedings of the subsequent experimentation is stated in Section 3. In section 4 the results are described and discussed. Finally, Future work and Conclusions are developed in Section 5.

1.1 Background

It was not until the 1970s when the first approach to music transcription was developed by Moorer [10] [11]. He was the first researcher to be convinced that a computer could detect, analyze and transcribe the pitches, chords and rhythm accents from a piece of music. His first investigations were focused on the development of vocal compositions transcriptions. Following his work, many researches diverged in new lines of investigation during the 1980s. Later, in the 1990s, Goto and Muraoka [6] worked in an algorithm for beat and rhythm tracking from and audio signal. After several years of investigation, both Goto and Muraoka were able to achieve decent polyphonic percussion tracks transcription. It was not until the 1990s when the first attempts to achieve polyphonic music transcriptors arose. *Unsupervised learning* became the state of art of AMT. In this methods, very few a prior assumption of the input signal were made, and they can be considered as the first step towards

the application of Deep Learning in Music Transcription applications [7]. Further information in traditional transcriptors can be found in [2],[1]

Throughout the last years, the interest in Neural Networks has grown considerably and so does the number of papers and publications focused on the possible applications of Deep Learning for music. One of these applications has gained significant importance over the last years: AMT. The main goal was to improve the best accuracies that were obtained using traditional Automatic methods and try to get closer to a professional musician transcription ability. The approaches were several, however, the aim of the current project is to experiment with different sets of Neural Networks in the application of Automatic Piano Music Transcriptions, and therefore, the publications of Sigtia [14] and [15] were of main interest. In [14], an Hybrid Recurrent Neural Network was proposed to be trained for Automatic music transcription using as input, the short-time Fourier transform (STFT) spectrogram of a set of Audio Files. The same principles and techniques were applied in [15] However, the objective in this case was to experiment and study the performance of a Convolutional Network for this duty. Also, instead of using the STFT, the Constant Q Transform(CQT) was used. Both innovations lead to significant improvements in the accuracy of the transcriptions.

1.2 Deep Neural Networks

Deep Neural Networks DNN are a sort of machine learning models that can be used for linear and non linear classification and regression tasks. The main characteristic of the DNNs is that they have several layers that are able to perform non linear transformations:

$$h_{layer+1} = f(W_{layer} h_{layer} + b_{layer}), \tag{1}$$

Where W_{layer} and b_{layer} are the Weight matrix and Bias vector respectively, $h_{layer+1}$ and h_{layer} the output values of the current layer and previous layer, h_{layer} and f is the activation function of the units in the layer. The parameters W_{layer} and b_{layer} are estimated using the backpropagation algorithm and Stocastic Gradient Descent (SGD). However, DNNs are designed for static data, not sequential. In the case of AMT, the time sequence must be included if we want to improve the performance of the transcriptor[15].

1.3 Recurrent Neural Networks

The Recurral Neural Networks(RNN) were conceived as a solution for the restricted ability of DNNs to handle sequential data. Due to this, RNNs are considered, a priori, a better option for AMT applications as consecutive frames will include both present and past features. In this case, the transformations done in each layer are:

$$h^t_{layer+1} = f(W^f_{layer} h^t_{layer} + W^r_{layer} h^{t-1}_{layer} + b_{layer}), \tag{2}$$

Where W^f_{layer} is the weight matrix of the forward pass of the layer, W^r_{layer} is the weight matrix for the recurrent connection and b_{layer} the Bias vector.The parameters W^f_{layer}, W^r_{layer} and b_{layer} are estimated using using the back propagation through time algorithm (BPTT) and SGD [15].

1.4 Long Short-Time Memory Networks

One of the main limitations of RNN is that they are not able to learn dependencies that are separated several time steps in time due to the well known issue of vanishing gradients[13]. They are just able to efficiently learn those dependencies that occurred in the previous step time. On the contrary, Long Short-Time Memory Networks (LSTM) are able to overcome this limitation. LSTM are a kind of RNNs architecture capable of learning long term dependencies by using the so called *memory cell*. This *memory cell* does not use any activation function within its recurrent components. Oppositely, the update step is done over 4 neural networks included in each *memory cell* commonly called gates.(See Fig.1) As a consequence, the stored value or state vector is not iteratively squashed over time, and the gradient does not tend to vanish during Back-propagation. Due to its main characteristics, LSTMs have replaced traditional RNNs for the majority of sequential/time series tasks.

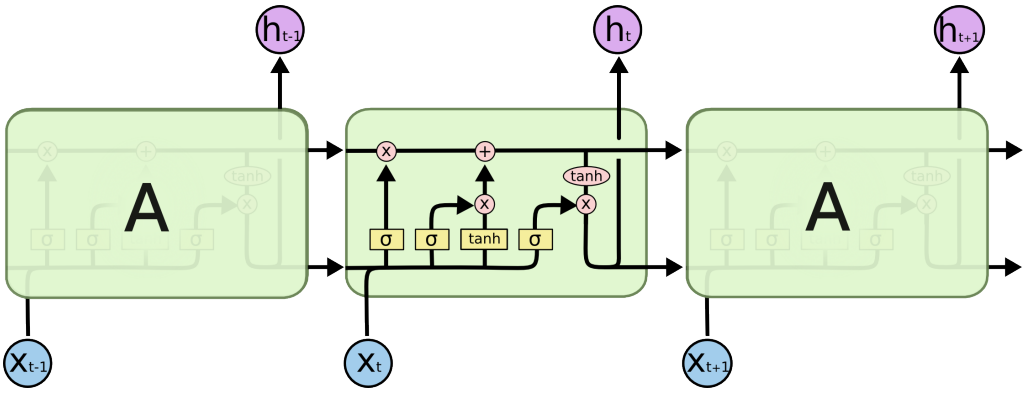


Figure 1: Individual Memory Cell of a LSTM network. In yellow, the main 4 neural network inside each LSTM unit, being from left to right: Forget gate, Input gate, tanh input cell and Output Gate

2 Method

2.1 Dataset Description

To accomplish the proposed goals of the current project the selection of the dataset was as important as the Neural Network structures chosen. In this case, MIDI Aligned Piano Sounds (MAPS) dataset was an optimal dataset for the experiments designed for the current project [4], [5]. The main advantage of these dataset is that every WAV file comes along with a text file with the different pitches duration and exact onset time. Therefore, label aligning was straight-forward. MAPS dataset is divided in 4 main sets of audio files: isolated notes and monophonic excerpts, chords with random pitch notes, usual chords from Western music and complete pieces of piano music. For the experiments arranged for this project, only the full musical piano pieces will be used. MAPS dataset includes 270 pieces of classical music from the main authors e.g.: Chopin, Mozart, Beethoven... These audio files are subdivided in 9 categories depending on the kind of piano and recording conditions, having 30 audio files and the corresponding transcriptions in each category, up to > 21 hours of music in total. Two of the categories were recorded in a real Disklavier piano, while the rest were recorded using different MIDI synthesizers.

2.2 Dataset Preprocessing

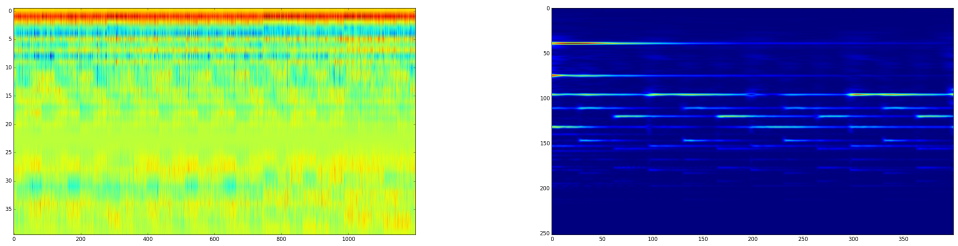
The first step was to divide the dataset in 3 subsets for training, validation and testing. For training, the 7 software-based categories were chosen. To ensure that the best possible performance of the networks was obtained, early stopping was going to be used. Therefore the selection of the validation data was very important as some of the songs of the dataset are repeated in several categories. For the validation data, 18 not repeated songs from the training data were chosen (700 mb of 7.7 gb of the entire training set). This way, the validation set was composed only of unseen songs. The objective of testing the network was not only studying the overall performance of the network over unseen data, but also its robustness under different conditions. As a consequence, two different test sets were arranged. The first one was designed to test the overall performance: the validation set and the unseen songs recorded in the real pianos were used (now on *Test set 1*). The second test set has the objective of testing not only the robustness of the network but also analyze the possible applications of the network as a real time AMT(now on *Test set 2*). As a consequence, this second test set was composed by the two categories that were recorded in the real piano, including both seen and unseen songs. This last set was also useful to determine the level of over-fitting in the trained network.

Next, the main features of the WAV audio files had to be extracted as training from raw audio data is considerably ineffective for this particular task. Historically, in AMT applications, the spectrum of the audio signal is used as the main features for the system. However, it was interesting to study first the performance of the Mel Filterbank Cepstrum Coefficients (MFCC) as the main features to use as the input for the networks as they have been rarely used in Music Transcription and they have proven to be optimal for Speech and Speaker recognition[9] and Music Genre Classification[8]. In order to compare and determine the adequacy of this features for the task of music transcription, parallel

experiments will be done using other kind of frequency spectrum features: Constant Q Transform features. This features are proposed in [15] as an optimal input representation for Automatic Music Transcription for two reasons: CQT is an optimal time-frequency representation for music signals, as the frequency axis is linear in pitch [3]. Furthermore, CQT features are dimensionally smaller than other frequency representations.

The audio files were individually transformed first from stereo to mono by computing the mean of the dual signal, and then MFCC features and Constant Q Transform were separately extracted from the mono audio signal. For the MFCC features, a 20ms window size was used along with a 10ms window separation. 40 filters were used with 40 coefficients. In the case of the CQT, and following the recommendations in [15], the audio signal was downsampled from 44.1kHz to 16kHz in order to reduce the amount of data that had to be transformed using CQT. The CQTs were computed over 7 octaves with 36 bins per octave and a hop size of 512 samples (32ms for 16kHz). As a result, 252 features per frame were obtained. In Fig. 2 there is a visual comparison between both transformations. We can see that for the CQT we get similar representation to a piano roll. We can conclude then that CQT is a priori a better option for AMT applications. In both cases, the data was represented as a matrix of dimensions *number of frames x number of features*. The frames of each song were concatenated together and split in individual files of 40000 frames per file.

The last step was data normalization. To do so, all the Training, Validation and Test sets were normalize within the limits of the training sets. Later, Training mean was calculated and subtracted from the three sets. Parallel to feature extraction, labeling alignment was computed. As each audio



(a) (b)

Figure 2: On the left: MFCC features of 30s of Piano Sonata No. 14 (Beethoven) song. On the right: CQT features of 30s of Piano Sonata No. 14 (Beethoven) song

file had a text file with the information about the pitch, its duration and onset time, time pairing the pitches with each frame was simple. The number of possible pitches were 88, and therefore, the label for each frame was represented as a vector of dimension 88, setting to one the pitches that were played in the current frame and zero otherwise. This vectors were stacked together and separated in files of 40000 thousand frames per file. This way, each CQT features file has its corresponding paired labels file.

2.3 Networks Structure

Due to time constraints, this project will focus only in two types of Networks: DNNs and LSTMs. It would have also been interesting to perform some experiments over a set of RNNs in order to compare its performance with a LSTM network, but RNNs experimentation under the same dataset and similar conditions is done in [15]. All the networks were built and trained using Keras with Tensorflow backend. The source code can be found in: `https://github.com/diegomorin8/Deep-Neural-Networks-for-Piano-Music-Transcription`

The basic principles of simple DNNs is already stated in Section 1. Four sets of DNN will be used for the experimentation: 1,2,3 and 4 hidden layers DNNs with 256 units in each hidden layer. Parameters tuning is out of the scope of this project due to time limitations, and Adam optimizer[12], which is a variant of the traditional Stochastic Gradient Descent (SGD), is used as in [15] it showed a correct performance on the proposed Networks. All the hidden layers used ReLU activation and the output layer's activation was set as Sigmoid as both the target output and Sigmoid function are bounded by

[0,1], being this function defined by:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

The input size depended on the number of input features. As a consequence, the networks designed for the MFCC features had an input size of 40 compared to the 252 input size that the networks designed for the CQT features. In both cases the output has to represent all the possible pitches, 88 in total, and therefore, the output layer size has to be of 88 units. Last but not least, Mean Squared Error (MSE) between the output and the labels vector for each frame was set as the loss function.

As in the case of the DNNs, four different structures of LSTM will be trained: 1,2,3 and 4 hidden layers LSTMs with 256 units in each hidden layer. In Section 1.4 it was explained how LSTMs are able to learn long term dependencies. It was important to determine how long this dependencies window/batch was going to be. 100 frames or time steps were chosen, being the window's size of 3.2 s. Doing this, the network would be able not only to classify the pitches in each frame, but also have into account important temporal dependencies such as global coherence, rhythm.... Then, each input matrix size was modified from *[number of frames, number of features]* to *[number of frames / 100, 100, number of features]*. Consequently, the input size of the network is *[100, number of features]*. As studied in [16], the optimal activation function for LSTM hidden units is the *hyperbolic tangent*. In the output layer, with size of 88 units, the activation function was set to be Sigmoid. For the same reason as with the DNNs, MSE was chosen as the loss function.In both the DNNs and the LSTMs, a dropout rate of 0.2 was added to avoid over-fitting.

2.4 Evaluation

To compute the evaluation, the output predictions were rounded: if the output values were lower than 0.5, they were rounded to zero, otherwise, the output was set to one. Later, the evaluation was frame-based, by calculating both the Average over the entire Test set, and the F measure. These parameters are calculated as:

$$\begin{aligned}
\text{Precision}(P) &= \sum_{t=0}^{N} \frac{TruePositives(t)}{TruePositives(t) + FalsePositives(t)} \\
\text{Recal}(R) &= \sum_{t=0}^{N} \frac{TruePositives(t)}{TruePositives(t) + FalseNegatives(t)} \\
\text{Accuracy}(A) &= \sum_{t=0}^{N} \frac{TruePositives(t)}{TruePositives(t) + FalsePositives(t) + FalseNegatives(t)} \\
\text{F-measure}(F) &= \frac{2PR}{P + R}
\end{aligned} \quad , \tag{4}$$

where N denotes the number of Frames in the data set that is being evaluated. Due to the characteristics of the dataset, it was really important to design a strategy to avoid over-fitting, as many songs were repeated in the different categories. Hence, after every epoch, the validation set was evaluated in order to stop the training if the evaluation F-measure did not improve for more than 15 epochs, avoiding this way possible over-fitting, as the validation set was composed by unseen songs. The weights were saved after every epoch. During testing, the same evaluation procedure was used.

2.5 Postprocessing

After the training was done, some cleaning of the predictions was performed. Typically, this postprocessing is done using complex but really effective techniques such as using and adaptive threshold during training to maximize the accuracy (instead of simply rounding the predictions) or training HMM models. However, in order to adapt to the time and job allocation constraints of this project, a simple post processing was designed. The main idea was to erase those predicted pitches which duration was smaller than the minimum duration of a pitch and filling small gaps between pitches.

3 Experiments

3.1 Audio signal features: MFCC and CQT

This first experiment had as main goal to perform an empirical comparison between MFCCs and CQTs as main features for AMT. There is no documentation on the possible applications of MFCC feaures in AMT, therefore, these features were compared in this first experiment with the CQT features, more common in AMT applications. To do so, two identical one hidden layer with 256 hidden units networks were used. The activation used was ReLU in the hidden later and sigmoid in the output layer. Early stopping was set as explained in Section 2.4.

3.2 Neural Network Structure: Deep Neural Networks and Long Short-Time Memory Networks

DNN's performance with sequential data has been proven to be very limited compared to other type of model structures. An example of a network that can handle, not only sequential dependencies but also long time dependencies are LSTM as described in Section 1.4.

In [14] and [15], different model structures are compared, such as DNNs, RNNs or Convolutional Networks. However, LSTM has not been studied yet for the task of Automatic Music Transcription. The current experiment was designed to compare the performance between traditional DNNs and LSTMs. To do so, 4 DNNs and 4 LSTMs were used, with 1,2,3 and 4 hidden layers respectively. All of them had 256 units in each hidden layer. The detailed explanation on both structures is reflected in Section 2.3. In all the cases, early stopping was done using the accuracy on the validation set in order to avoid possible over-fitting. All the networks were tested in the trained networks using both the Test set 1 and 2 (see Section2.2). During this experiment, post processing is computed, and the final performance after post processing is compared to the performance obtained before this cleaning operation.

4 Results

4.1 Audio signal features: MFCC and CQT

After training two identical networks with the MFCC and CQT features respectively, the Accuracy and F-measure of the predictions over the First Test set were computed and summarized in Table.1. Even though a more exhaustive experimentation on this comparison should be done to completely discard MFCC features as an optimal feature for Music Transcription, on the following experiments MFCC was not considered as the obtained accuracy over the Test set is significantly lower when MFCC features were used.

Table 1: MFCC and CQT features accuracies. DNN with 1 hidden layer of 256 units. ReLU activation

Feature Type	F-measure	Accuracy
MFCC	3.46%	1.76%
CQT	**62.89%**	**45.88%**

4.2 Neural Network Structure: Deep Neural Networks and Long Short-Time Memory Networks

In this case, after all the networks were trained, they were tested using the First Test set. The computed Accuracies and F-measures are shown in the Table 2.

We can observe that in the case of the DNNs the best results are obtained by the network with 3 hidden layers. The LSTM performance is quite similar to the one obtained by the DNNs, contrary to what happened with the RNN networks in [15] that improved the accuracy of the DNNs. Either way, as expected the post-processing operation did not improve the accuracies significantly. The method used is very inefficient compared to other sophisticated methods as Hidden Markov Models (HMM).

Table 2: Accuracies and F-measures comparison table for different networks, after testing using the First Test set.

			Predicted			Post Processed	
Model	Layers	Units	F-measure	Accuracy	epoch	F-measure	Accuracy
DNN	1 layer	256 units	69.13%	52.83%	100	70.45%	54.37%
LSTM	1 layer	256 units	65.80%	49.029%	203	66.07%	49.34%
DNN	2 layer	256 units	68.61%	52.22%	45	69.84%	53.65%
LSTM	2 layer	256 units	68.55%	52.15%	99	69.01%	52.68%
DNN	3 layer	256 units	**69.36%**	**53.09%**	111	**70.61%**	**54.58%**
LSTM	3 layer	256 units	68.95%	52.61%	98	69.36%	53.09%
DNN	4 layer	256 units	68.78%	52.42%	154	69.99%	53.83%
LSTM	4 layer	256 units	64.05%	47.11%	178	64.52%	47.62%

The same trained networks, were tested using the Second Test set to test the performance of the network when a real recorded audio of a piano was used to check the robustness of the network. Also, as some of the tracks in this test set were included in the training set, it was indeed a good method to test the level of over-fitting in the trained network. The results of this testing are shown in the table 3.

Table 3: Accuracies and F-measures comparison table for different networks, after testing using the First Test set

			Predicted			Post Processed	
Model	Layers	Units	F-measure	Accuracy	epoch	F-measure	Accuracy
DNN	1 layer	256 units	64.41%	47.50%	100	65.60%	48.81%
LSTM	1 layer	256 units	61.77%	44.68%	203	61.84%	44.76%
DNN	2 layer	256 units	64.54%	47.65%	45	65.77%	48.96%
LSTM	2 layer	256 units	63.23%	46.23%	99	63.48%	46.50%
DNN	3 layer	256 units	65.29%	48.47%	111	**66.54%**	**49.86%**
LSTM	3 layer	256 units	**66.05%**	**49.31%**	98	66.37%	49.67%
DNN	4 layer	256 units	65.29%	48.47%	154	66.53%	49.85%
LSTM	4 layer	256 units	63.75%	46.79%	178	64.17%	47.24%

We can observe that in the case of the LSTMs the best results are obtained by the network with 3 hidden layers. However, after post-processing is the DNN with 3 hidden layer the Network with better performance. Comparing Tables 2 and 3 is easy to observe that using the Second Data set for testing arises worse accuracy results. This can be understood as a low level of over-fitting in the network along with sufficient robustness against changes in the recording environment. However, the results of this second testing are quite similar to the ones obtained for DNNs networks in [15] using a similar set of parameters, being this a positive indicator on the method used for the current project.

In figure. 3 an example of the predicted sequence before and after post-processing is shown in the format of a piano roll along with the ground truth sequence and the prediction from a 3 layer LSTM network.

5 Discussion and Conclusions

The first objective of the project was to check the possible applications of MFCC for AMT. Regarding the results shown in the Table 1 it can be concluded that MFCC features are significantly less optimal for AMT applications than CQT features.

Besides, we can observe that the results obtained after testing the final networks with the Second Test set on the DNNs are really similar to [15]. This investigation paper was used as a reference to build and train the DNNs. Therefore, these DNNs and the RNNs tested in [15] can be used as a comparison reference for the LSTM. Between this three models of networks, the RNNs are the ones that have a higher performance. In pitching classification, local coherence is more important than

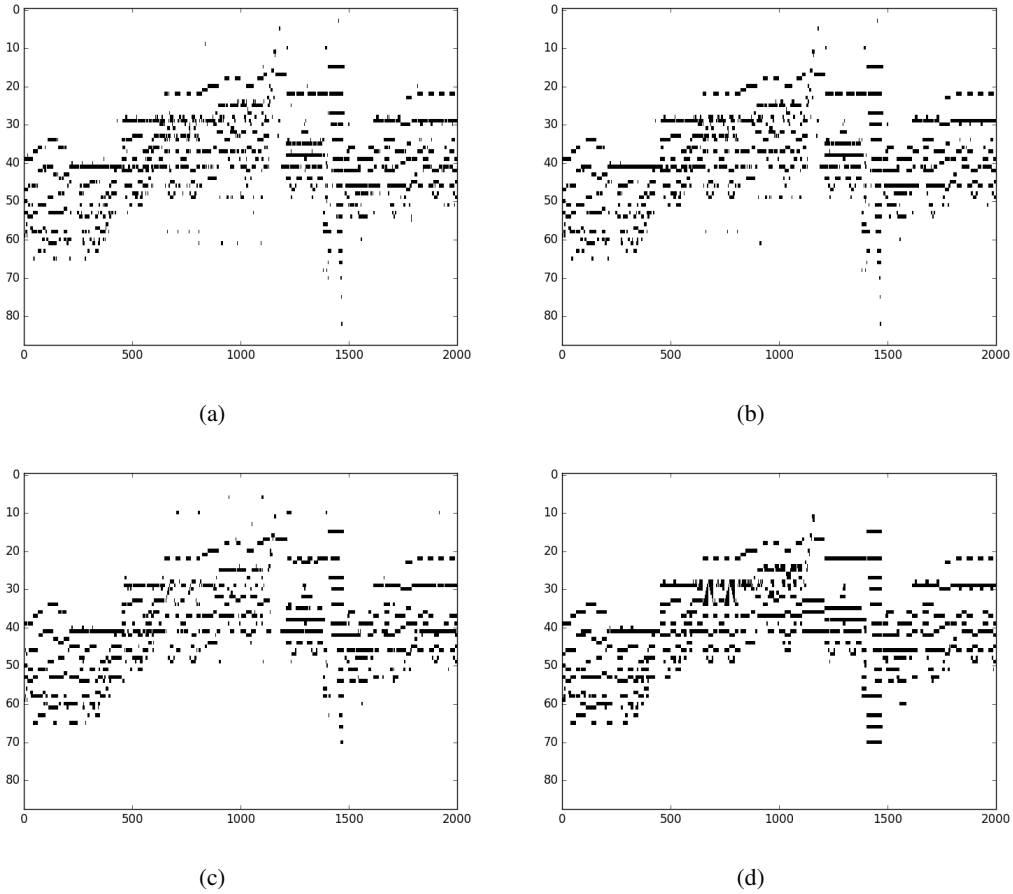


Figure 3: (a) Rounded prediction for the 3 hidden layer DNN of an unseen piece of audio file (b) Post-processed prediction (c) Rounded prediction of the 3 hidden layer LSTM(d) Ground truth of the piece of audio file

global features. LSTM will perform better, for instance, in music generation tasks. The comparisopn between DNNs and LSTMs is, however, not evident: both of them have different drawbacks. We can observe in Figure. 3(b) and (c) that while DNNs can identify the pitches better than LSTM they estimate the duration of the pitches much worse than the LSTM (note that Figure. 3(c) was not post-processed). Hence, it will be interesting to test an hybrid DNN-LSTM set to complement their advantages reducing this way the main withdraws of each method. This can become the most probable and interesting future line of investigation.

Last but not least, the early stopping strategy worked correctly as the networks did not suffered from severe over-fitting during training as we can observe that the accuracy for the Second Test set is similar to the one obtained for completely unseen data. Also, as the network was trained on classical music piano pieces, it is more likely to perform better with this type of music.

To have a further understanding of the results, a simple code to transform the predicted transcriptions to a MIDI music was written. Different commonly known songs along with some examples from the Test Set 1 where inputed to the network and the predicted were transformed back to MIDI file and the results were gathered in `https://goo.gl/U1FKnr`. In this examples, some of the artifacts that appear in the predicted transcriptions are Classical music alike. Therefore, as another future line of work, it would be interesting to train the network using the entire MAPS dataset, including individual pitches, chords and scales to reduce this artifacts and improve the accuracy.

References

[1] F. Argenti, P. Nesi, and G. Pantaleo. Automatic music transcription: from monophonic to polyphonic. Department of Systems and Informatics, University of Florence.

[2] Emmanouil Benetos, Simon Dixon, Dimitrios Giannoulis, Holger Kirchhoff, and Anssi Klapuri.

[3] Judith C. Brown. Calculation of constant q spectral transform.

[4] Badeau R. Emiya, V. Emiya and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle, 2010. IEEE Transactions on Audio, Speech and Language Processing, (to be published);.

[5] V. Emiya. *Transcription automatique de la musique de piano*. PhD thesis, elecom Paris-Tech, 2008.

[6] Masakata Goto and Yoichi Muraoka. A beat tracking system for acoustic signals of music. *ACM International Conference on Multimedia*, pages 365–372, 1994.

[7] Anssi Klapurri. Introduction to music transcription. Technical report, Institute of Signal Processing, Tampere University of Technology, Korkeakoulunkatu 1, 33720 Tampere, Finland, 2006.

[8] Gursimran Kour and Neha Mehan. Music genre classification using mfcc, svm and bpnn. *International Journal of Computer Applications (0975 ˘ 8887)*.

[9] Pavel Matǰka, Ondřej Glembek, Ondřej Novotnỳ, Oldřich Plchot, František Grèzl, Lukàš Burget, and Jan "Honza" Černockỳ. Analysis of dnn approaches to speaker identification, 2016. Brno University of Technology, Speech@FIT and IT4I Center of Excellence, Brno, Czech Republic.

[10] J.A. Moorer. *On the Segmentation and Analysis of Continuous Musical Sound by Digital Computer*. PhD thesis, On the Segmentation and Analysis of Continuous Musical Sound by Digital Computer, 1975.

[11] J.A. Moorer. On the transcription of musical sound by computer. *Computer Music Journal*, 2(1):7–11, 1977.

[12] Diederik P. and Jimmy Lei Ba. Adam: A method for stochastic optimization, 2015. 3rd International Conference for Learning Representations, San Diego.

[13] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks, 2013. Université de Montréal, 2920, chemin de la Tour, Montréal, Québec, Canada, H3T 1J8, Speech@FIT, Brno University of Technology, Brno, Czech Republic.

[14] Siddharth Sigtia, Emmanouil Benetos, Nicolas Boulanger-Lewandowski, Tillman Weyde, Artur S. d'Avila Garcez, and Simon Dixon. A hybrid recurrent neural network for music transcription. arXiv:1411.1623v1, 2014. Centre for Digital Music, EECS, Queen Mary University of London, London, UK; Department of Computer Science, City University London, London, UK ; Dept. IRO, Universite de Montreal, Montreal (QC), H3C 3J 7, Canada.

[15] Siddharth Sigtia, Emmanouil Benetos, and Simon Dixon. An end-to-end neural network for polyphonic piano music transcription. arXiv:1508.01774v2, 2014. Centre for Digital Music, School of Electronic Engineering and Computer Science, Queen Mary University of London, E1 4NS, London, U.K.

[16] Martin Sundermeyer, Ralf Schluter, and Hermann Ney. Lstm neural networks for language modeling, 2012. Human Language Technology and Pattern Recognition, Computer Science Department, RWTH Aachen University, Aachen, Germany.