Deep Neural Networks for Piano Music Transcription

Overview

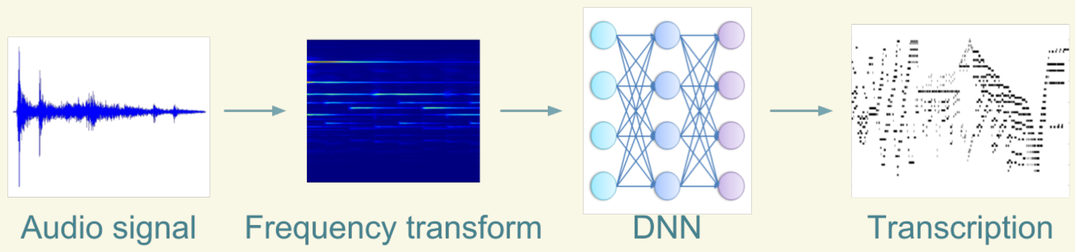
- Review of the different approaches for Automatic Polyphonic Piano Music Transcription.
- Experimentation with different features extraction methods - **MFCC and CQT**.
- Test different sets of **DNN and LSTM** for comparison.
- Piano recordings in different environmental conditions for robustness study.

Background

Moorer - 1975: First **computer-based** vocal compositions automatic transcriptor.

2000s: **STFT analysis and HMM-based** post-processing.

Sigtia - 2015: Proposed several **End-to-end Neural networks** approaches for Automatic Music Transcription



MIDI Aligned Piano Sounds (MAPS) Dataset

- WAV, MIDI and text files with pitch annotation of each song.
- 270 classic piano pieces(> 21 hours).
- 9 different recording environments

Training set: **7 software-based recording environments**.

Validation: 18 unseen audio files from the training set.

Test set 1: 30 unseen audio files from all the environments .

Test set 2: designed for robustness and over-fitting check - all the files recorded in a **real piano.**

Data pre-processing

- Down-sampling: **44.1kHz to 16kHz**
- Feature extraction:

MFCC: - 20ms window size - 10ms window separation - **40 coefficients**

CQT: - 7 octaves - 36 bins per octave - hop size of 32 ms - **252 features**

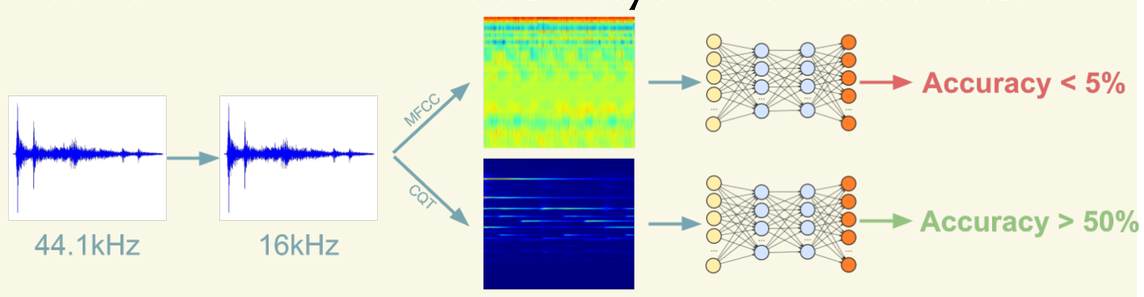
- **Pitch aligning** using a custom algorithm.

Experiments

- **MFCC vs CQT** features

Objective: MFCC and CQT features accuracy comparison.

Network: **DNN - 1 hidden layer with 256 units.**



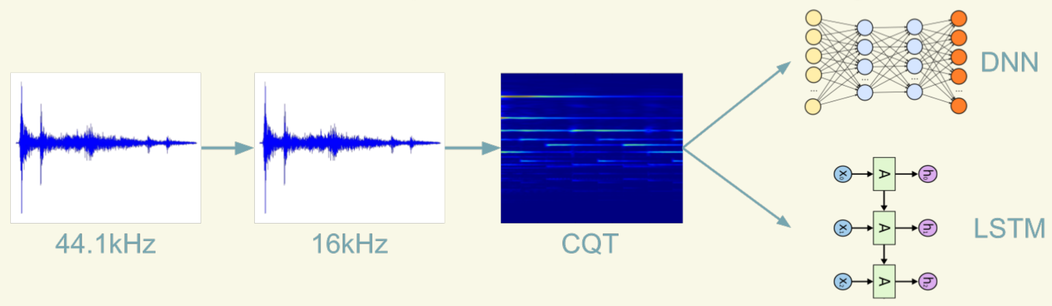
- **DNN vs LSTM**

Objective: Performance comparison between different type of networks and sizes.

Features: **CQT** - much better results in the first experiments.

Network A: **DNN** - {1,2,3,4} hidden layers with 256 units. Hidden layer: **ReLU** - Output: **Sigmoid**

Network B: **LSTM** - {1,2,3,4} hidden layers with 256 units. Hidden layer: **tanh** - Output: **Sigmoid**



- Data **post-processing**: Simple algorithm to clean small artifacts in the predictions.

Results

Training: Using **Keras** with Tensorflow backend:

- **Adam** optimizer
- **20% dropout** to avoid over-fitting.
- **Early stopping** using Validation set.
- Best results:

			Predicted		Post Processed	
Model	Size	Test Set	F-measure	Accuracy	F-measure	Accuracy
DNN	3L	Set 1	**69.36%**	**53.09%**	**70.61%**	**54.58%**
LSTM	3L	Set 1	68.95%	52.61%	69.36%	53.09%
DNN	3L	Set 2	65.29%	48.47%	**66.54%**	**49.86%**
LSTM	3L	Set 2	**66.05%**	**49.31%**	66.37%	49.67%

Output example: 1min 30s piece from Test set 1.

Diego González Morín
diegogm@kth.se
KTH - Kungliga Tekniska Högskolan