End-to-end polyphonic piano transcription

Patrick Saux

Audio Signal Analysis

March 17th 2020

Table of Contents

Piano transcription

- Musician: from notes indication (e.g sheet music) to sound.

Piano transcription

- Musician: from notes indication (e.g sheet music) to sound.
- Transcription: from sound to notes, human annotation.

Piano transcription

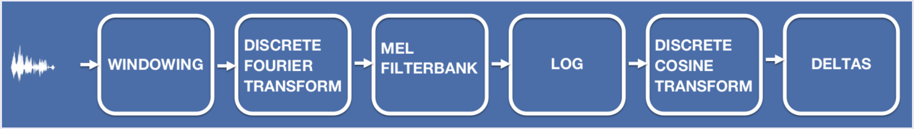
- Musician: from notes indication (e.g sheet music) to sound.
- Transcription: from sound to notes, human annotation.
- Automatic Music Transcription: do it algorithmically with nothing else than raw sound input.

Piano transcription

- Musician: from notes indication (e.g sheet music) to sound.
- Transcription: from sound to notes, human annotation.
- Automatic Music Transcription: do it algorithmically with nothing else than raw sound input.
- In this paper: (almost) end-to-end supervised learning.

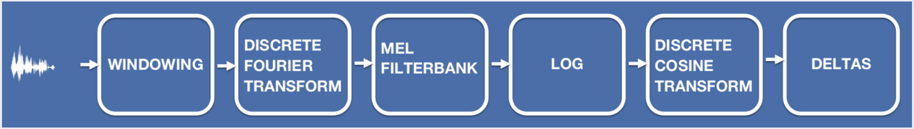
Input

Definition (MFCC)



Input

Definition (MFCC)



Why MFCC?

- Log scale perception, pitch-invariant patterns,
- Source-filter separation,
- Low dimensional,
- Robust to small deformation.

Dataset

MAPS dataset.
Input: downsample 4x, 7 octaves, 36 bins per octave :

$$x(t) = MFCC(t) \in \mathbb{R}^{252}.$$

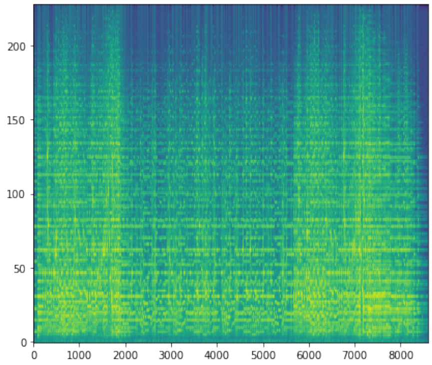
Output: 12 notes per octave:

$$y(t) \in \{0, 1\}^{88}.$$

Sequence length: $t = 0, \ldots, T$, $T \approx 1000$.

Dataset

MAPS_MUS-chpn_op66_AkPnBcht.wav



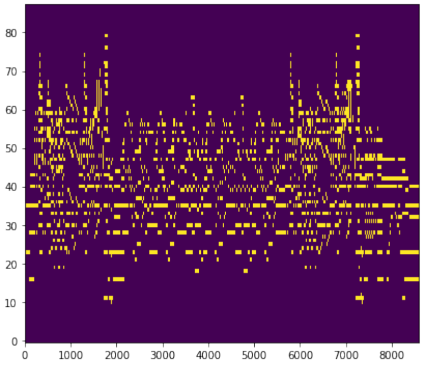


Table of Contents

Acoustic vs Language models

Why not learn directly $x(t) \mapsto y(t)$?

Acoustic vs Language models

Why not learn directly $x(t) \mapsto y(t)$?

- Sequence to sequence : frames are not independent.
- More efficient way to extract information.

Acoustic vs Language models

Why not learn directly $x(t) \mapsto y(t)$?

- Sequence to sequence : frames are not independent.
- More efficient way to extract information.
- Bayes rule:

$$\mathbb{P}(y_0^t | x_0^t) \propto \mathbb{P}(y_0^{t-1} | x_0^{t-1}) \mathbb{P}(y_t | y_0^{t-1}) \mathbb{P}(y_t | x_t)$$

Acoustic vs Language models

Why not learn directly $x(t) \mapsto y(t)$?

- Sequence to sequence : frames are not independent.
- More efficient way to extract information.
- Bayes rule:

$$\mathbb{P}(y_0^t|x_0^t) \propto \mathbb{P}(y_0^{t-1}|x_0^{t-1}) \underbrace{\mathbb{P}(y_t|y_0^{t-1})}_{\text{language}} \underbrace{\mathbb{P}(y_t|x_t)}_{\text{acoustic}}$$

Acoustic vs Language models

Why not learn directly $x(t) \mapsto y(t)$?

- Sequence to sequence : frames are not independent.
- More efficient way to extract information.
- Bayes rule:

$$\mathbb{P}(y_0^t|x_0^t) \propto \mathbb{P}(y_0^{t-1}|x_0^{t-1}) \underbrace{\mathbb{P}(y_t|y_0^{t-1})}_{\text{language}} \underbrace{\mathbb{P}(y_t|x_t)}_{\text{acoustic}}$$

- Language model can be trained on large MIDI corpora.

Acoustic models

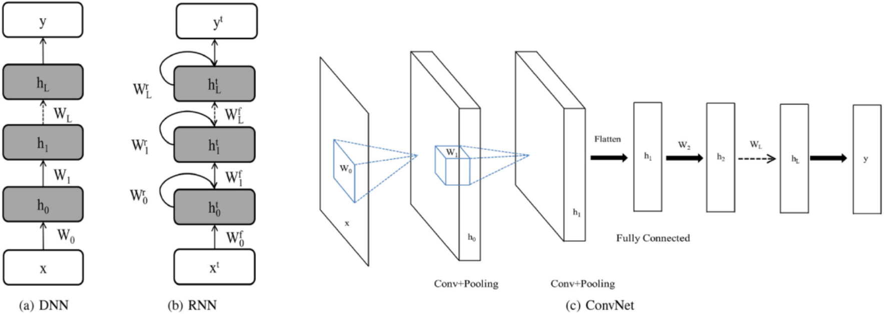


Fig. 1. Neural network architectures for acoustic modelling.

Language models

Definition (RNN-NADE)

The language model is

$$\mathbb{P}(y_t | y_0^{t-1}) = \sigma(V_t h_t + b_v^t)$$

where

$$\begin{aligned} h_t &= \sigma(W_{:,<t} y_0^{t-1} + b_h^t) \\ b_v^t &= b_v + W_1 h_t^{RNN} \\ b_h^t &= b_h + W_2 h_t^{RNN} \\ h_t^{RNN} &= \text{hidden state of final RNN layer} \end{aligned}$$

Language models

Definition (RNN-NADE)

The language model is

$$\mathbb{P}(y_t | y_0^{t-1}) = \sigma(V_t h_t + b_v^t)$$

where

$$\begin{aligned} h_t &= \sigma(W_{:,<t} y_0^{t-1} + b_h^t) \\ b_v^t &= b_v + W_1 h_t^{RNN} \\ b_h^t &= b_h + W_2 h_t^{RNN} \\ h_t^{RNN} &= \text{hidden state of final RNN layer} \end{aligned}$$

Tractable : full sequence likelihood gradient w.r.t parameters known in closed-form

Table of Contents

Decoding

Predicted sequence is $\hat{y} = \arg\max_y \mathbb{P}(y|x)$.

- Exact inference on graphical model (Viterbi) : intractable (2^{88} configurations *per frames*).

Decoding

Predicted sequence is $\hat{y} = \arg\max_y \mathbb{P}(y|x)$.

- Exact inference on graphical model (Viterbi) : intractable (2^{88} configurations *per frames*).
- Greedy decoding $\hat{y_t} = \arg\max_{y_t} \mathbb{P}(y_t|y_0^{t-1})\mathbb{P}(y_t|x_t)$: very suboptimal

Decoding

Predicted sequence is $\hat{y} = \arg\max_y \mathbb{P}(y|x)$.

- Exact inference on graphical model (Viterbi) : intractable (2^{88} configurations *per frames*).
- Greedy decoding $\hat{y_t} = \arg\max_{y_t} \mathbb{P}(y_t|y_0^{t-1})\mathbb{P}(y_t|x_t)$: very suboptimal
- In between: **beam search**.

Decoding

Two refinements to standard beam search:

- **Branching**: at each iteration, draw only the K best candidates from $\mathbb{P}(y_t|x_t)$ to integrate the beam.

Decoding

Two refinements to standard beam search:

- **Branching**: at each iteration, draw only the K best candidates from $\mathbb{P}(y_t|x_t)$ to integrate the beam.
- **Locally sensitive hash**: prune candidates with similar likelihood to avoid beam saturation.

Table of Contents

Own (limited) experiments

Implementation in PyTorch.
Vanilla model:

- **Acoustic model** : feed-forward, 4 layers of 512 neurons, 25% dropout, ReLU, Adam optimizer.
- **Language model**: LSTM, 2 layers of 128 neurons, Adam optimizer.
- **Decoding**: greedy.
- Out-of-sample F score: 42%.

Stacked frames:

- Same but reads 3 frames of MFCC to predict the central transcription frame.
- Out-of-sample F score: 53%.

Paper results

- **Acoustic model** : CNN, 2 layers.
- **Language model**: RNN-NADE.
- **Decoding**: Hashed branching beam search.
- Out-of-sample F score: 74%.

Hashed branching beam search :

- Requires narrower beam.
- Faster: 22 minutes vs 20 hours on the test set.
- Better decoding accuracy.

Chopin transcription

MAPS_MUS-chpn-p10_AkPnStgb.wav