A comparison of feed forward neural network architectures for piano music transcription

Matija Marolt
Faculty of Computer and Information Science
University of Ljubljana, Tržaška 25
1000 Ljubljana, Slovenia
matija.marolt@fri.uni-lj.si

Abstract

This paper presents our experiences with the use of feed forward neural networks for piano chord recognition and polyphonic piano music transcription. Our final goal is to build a transcription system that would transcribe polyphonic piano music over the entire piano range. The central part of our system uses neural networks acting as pattern recognisers and extracting notes from the source audio signal. The paper presents results obtained by using several feed forward neural network architectures for transcription, namely multilayer perceptrons, RBF networks, support vector machines and time-delay networks.

1. Introduction

Music transcription could be defined as an act of listening to a piece of music and writing down music notation for the piece. We could in a way compare it to speech recognition, where an audio signal is converted to phonemes, syllables and finally words; music transcription converts an audio signal to notes and determines their starting times, duration and loudness. Speech recognition has recently been quite successfully solved (at least for single words) and we see more and more applications coming to the market. For polyphonic music transcription this is not yet the case.

Recently, there have been several reports on systems for transcription of polyphonic music: [Martin, 1996] for example, uses a blackboard system for piano music transcription, several other systems include [Nunn, 1997], [Rossi, 1998] and [Klapuri, 1998].

In our approach, we employ artificial neural networks as the main part of our transcription system. Artificial neural networks have been used for speech recognition and other pattern recognition tasks for a long time. They are especially suitable for such tasks because of their ability to learn from examples, generalise and robustness to noise. Because of this, it is our main motivation to study the usability of these algorithms for transcription. We present preliminary results obtained by using feed forward neural networks for piano music transcription, where by transcription we mean recognising the note and time when the note occurred (length and dynamics are not taken into consideration).

2. Our Approach

We have built a very simple system for transcription of piano music. The system attempts to correctly determine the notes and their starting times in a polyphonic piano performance. It has three main stages described below.

2.1 Preprocessing

The first stage of the system transforms the input audio signal into time-frequency space. To perform the transformation, we chose a simple correlation-based transformation, which enables us to arbitrarily pick the frequency and time resolution of each of the transformation's frequency bands. We chose a logarithmical division of frequency bands with spacing of one semitone for lower frequencies and up to 1/4 semitone for higher frequencies. The lowest and highest frequency bands were located at 50 and 9000 Hz respectively. Time resolution of bands ranged from 80 ms at lower frequencies to 10 ms at higher frequencies. Hamming window was used for windowing. We compared several transformations with different numbers of frequency bins and with different TF parameters and finally settled for a transformation with 304 frequency bins.

2.2 Transcription

The main part of the system, that actually performs the transcription, consists of a set of 88 feed forward neural networks - one network for each piano note (A1-C9 MIDI notation). The input of each network consists of one or more TF transformed and normalised time frames obtained from the preprocessing stage. Each network has only one output. An output value of over 0.5 (training threshold) means that the target note (the one the network was trained to recognise) is present in the input, a value below 0.5 means, that the note is not present.

2.3 Postprocessing

Inputs of the postprocessing stage are activations of all 88 neural networks representing the occurrence of piano notes in the input audio signal. These inputs are therefore numbers from 0 to 1, indicating how each network classified its input. Postprocessing currently consists of a very simple time averaging algorithm to reduce noise and prevent a single high neural network
activation to cause the system to mark a note as present. Several consecutive high neural network activations are needed for a note to be declared present in the input audio signal. The final output of the system are notes and their starting times.

3. Results

3.1 Neural Networks and Training

We have tested four feed forward neural network architectures: multilayer perceptrons, radial basis function networks, support vector machines and time-delay networks.

Networks were trained on a database of approximately 400,000 piano chords. To build the database, we first gathered samples of single piano notes covering the whole piano range (A1 to C9) at different intensity levels. Samples were taken from several synthesiser patches and commercially available piano sample CD-ROMs. We then constructed the chords (polyphony from one to six) by mixing these sampled piano notes.

Each network has been trained to recognise the presence of one piano note (target note) in the input chord. The training set for each network included approx. 30000 chords with 1/3 of chords containing the target note. A time-frequency transformation was used (see section 2.1) to extract one or more time frames from each chord. These time frames were normalised and used for supervised learning of the networks; each network has been trained to an output of 1 if its target note was present in the chord or 0 if not.

3.2 Chord Recognition

Because networks were trained on individual chords, we first tested their performance on a test database containing chords not used for training. Each network was tested on approximately 5000 chords different from the ones used for training (we also used several new piano samples).

Multilayer perceptrons (MLP), as the most widely used neural network architecture, achieved very good results on the test set. We trained and tested MLP networks with different numbers of neurons in the hidden layer and finally settled on 18. Less neurons resulted in worse results especially for lower octaves, while for higher octaves a smaller number of neurons would be sufficient.

MLPs were trained and tested with three types of inputs; the first included one time frame from each chord with all frequency bins of the TF transformation, which resulted in 304 input neurons. The second included one “compressed” time frame with only frequencies around notes that hadpartials, that interfered with partials of the target note; this resulted in 149 input neurons (for first six interfering partials). The third included two consecutive “compressed” time frames (same as above) and resulted in 298 input neurons.

Networks with 304 and 298 input neurons produced very similar results. For both, note accuracy for lower octaves (A1-E3) averaged to 94%, E3-A4 around 97% and above A4 to 98.5%. Better performance for higher octaves is due to better frequency resolution. The performance of networks with 149 input neurons was around 1% worse. Octave errors were the most common cause of errors, especially in the middle octaves.

Time-delay neural networks (TDNNs — see [Waibel, et al., 1989]) were the only tested networks that use time as an implicit parameter (are time-aware). They produced the best results. We trained and tested TDNNs with two and three consecutive time frames at the input layer (608 and 912 input neurons respectively). Results for both networks were very similar and almost 1% better than results of MLP networks with 304 and 298 input neurons. TDNNs with “compressed” inputs performed a little worse, but still better than MLP networks.

Support vector machines (SVMs — see [Hearst, et. al., 1998]) are a relatively new class of machine learning algorithms. We trained SVMs using several types of kernels and got the best results with standard rbf kernels. Results were comparable to those obtained by TDNN networks.

Radial basis function (RBF) networks have also been tested, but their performance was not so good. We used networks with 50 center vectors and test results were approximately 5% worse than MLP results. Results could probably be improved by increasing the number of center vectors, but training time then slowly becomes too large.

3.3 Music Transcription

We have also tested the trained MLP and TDNN networks in the context of the system described in section 2. We used several MIDI files of solo piano performances and rendered excerpts from them with different piano samples. The pieces ranged from very simple Bach’s Two-part Inventions to more complex excerpts from Tchaikovsky’s Nutcracker Suite.

Transcription results obtained by using MLP networks with 304 input neurons are given in table 1. Results are given for transcriptions of excerpts of five pieces: J.S. Bach’s Two-part Invention No. 1 (BWV 772), Bach’s Three-part Sinfonia No. 1 (BWV 787), Bach’s English Suite No. 1 (BWV 806), Tchaikovsky’s Nutcracker Suite Miniature Overture and Nutcracker Suite Waltz of the Flowers.

The second and third columns of table 1 represent the average and maximum polyphony of transcribed excerpts. The fourth column (notes found) represents the percentage of notes in each piece that were correctly transcribed. The fifth column (extra notes)
represents the number of additional notes that were found, but were not present in the input.

<table>
<thead>
<tr>
<th>piano piece</th>
<th>av. poly</th>
<th>max poly</th>
<th>notes found</th>
<th>extra notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 pt. inv.</td>
<td>2</td>
<td>4</td>
<td>96.6</td>
<td>19.4</td>
</tr>
<tr>
<td>3 pt. sin</td>
<td>2.7</td>
<td>3</td>
<td>99.8</td>
<td>19.0</td>
</tr>
<tr>
<td>English suite</td>
<td>3</td>
<td>4</td>
<td>96.9</td>
<td>25.6</td>
</tr>
<tr>
<td>Nutcrack. ovr.</td>
<td>3.1</td>
<td>6</td>
<td>96.1</td>
<td>21.8</td>
</tr>
<tr>
<td>Nutcrck. waltz</td>
<td>5</td>
<td>15</td>
<td>94.6</td>
<td>34.9</td>
</tr>
</tbody>
</table>

Table 1: MLP transcription results

The first thing we notice is that there is a large number of extra notes present in the transcribed output. This number gets larger as polyphony increases. Analysis showed that most of these errors are octave (or similar) errors, where strong partials of notes in the input signal cause new extra notes to be detected by neural networks.

![Figure 1: typical errors](image)

An example of such errors can be seen in figure 1, which represents a short part of transcription of Waltz of the Flowers. Two errors are marked: the first G6 note is an octave error, while the C2 note also represents a common mistake; many times very short low-pitched notes appear in the score. We contribute these errors to the fact that networks for lower pitched notes can not be so accurately trained.

The percentage of input notes correctly transcribed is quite high and does not deteriorate too much when polyphony increases. A problem that appears is that starting times of notes are sometimes transcribed later than notes actually occur. We believe that these errors occur due to the fact that the networks were trained only on steady portions of chords in the training set. The attack portion was left out, so networks react unpredictably until the sound of each note settles down to its steady portion.

Transcription results form MLP networks with 298 input neurons (two consecutive compressed time frames) were similar, but with more extra notes, while MLPs with 149 input neurons (one compressed time frame) performed much worse.

TDNNs with 608 inputs (two full time frames) were slightly better; the percentage of found notes was approximately the same, while the number of extra notes was smaller especially in pieces with lower polyphony.

4. Summary and Future Work

The presented system gives good results, when used for chord recognition, but does not perform so well when it comes to transcribing polyphonic music. Especially the number of extra notes produced by the system is higher than desired.

The current system is very simple and several things could be improved:

- In the current system, networks are trained on a different domain (chords) than they are used on (transcription). By constructing a training database from piano performances, we hope to improve the performance and decrease the number of extra notes.
- A better postprocessing stage could be implemented. It could take into consideration not only neural networks' outputs, but also other factors, such as information obtained directly from the audio source (onset detection, partials,...). This could reduce several types of errors, including extra notes and incorrect starting times.
- We will evaluate other time-aware neural network models, such as locally recurrent networks.
- We also plan to evaluate other TF transformations, such as CASA based transformations.

References


