A Neural Organist
improvising baroque-style melodic variations

Dominik Hörnel
Peter Degenhardt

Institut für Logik, Komplexität und Deduktionssysteme,
Universität Karlsruhe, D-76128 Karlsruhe, Germany
dominik@ira.uka.de, http://i11www.ira.uka.de

Abstract

We present a multi-scale neural network system producing melodic variations in a style directly learned from music pieces of composers like Johann Sebastian Bach and Johann Pachelbel. Given any melody, the system first invents a four-part chorale harmonization and then improvises a variation of any chorale voice. Unlike earlier approaches to the learning of melodic structure, the system is able to learn and reproduce high-order structure like harmonic, motif and phrase structure in melodic sequences. This is achieved by using mutually interacting neural networks operating at different timescales, in combination with an unsupervised learning mechanism to classify and recognize musical structure. The results are complete music pieces, e.g. in the style of chorale partitas written by J. Pachelbel. Their quality has been judged by experts to be comparable to improvisations invented by an experienced human organist.

1 Introduction

The art of melodic variation has a long tradition in Western music. Almost every great composer has written music pieces inventing variations of a given melody, e.g. Mozart's famous variations KV 265 on the melody "Ah! Vous dirai-je, Maman", also known as "Twinkle twinkle little star". At the beginning of this tradition there is the baroque type of chorale variations. These are organ or harpsichord variations of a chorale melody composed for use in the Protestant church. A prominent representative of this kind of composition is J. Pachelbel who wrote about 50 chorale variations or partitas on various chorale melodies. At his time he was known as "a perfect and rare virtuoso" whose works influenced many other composers such as J. S. Bach. Musical practice had a lot of impact on Pachelbel's style of composition. Indeed most of his chorale partitas can be seen as improvisations of an organist who invents 'real-time' harmonizations and melodic variations of a given chorale melody. This way of composing is very similar to the behavior of our neural network system. The problem of learning melodic variations was already studied in [3]. Although the approach produced some musically convincing local sections, the results lacked global coherence in general. The neural network model we present here is able to learn global structure from music examples by using two mutually interacting neural networks that operate at different timescales. The main idea of the model is a combination of unsupervised and supervised learning techniques to perform the given task. Unsupervised learning classifies and recognizes musical structure, supervised learning is used for prediction in time. The model has been tested on simple children-song melodies in [4]. In the following we will illustrate its practical application to a complex musical task — the learning of melodic variations in the style of J. Pachelbel.

2 Task Description

Given a chorale melody, the learning task is achieved in two steps:

1. A chorale harmonization of the melody is invented.
2. One of the voices of the resulting chorale is chosen and provided with melodic variations.

Both subtasks are directly learned from music examples composed by J. Pachelbel and performed in an interactive composition process which results in a chorale variation of the given melody. The first task is performed by HARMONET [5], a neural network system which is able to harmonize melodies in the style of various composers like Bach and Pachelbel. The second task is performed by the neural network system presented in the following. For simplicity we have considered melodic variations consisting of 4 sixteenth notes for each melody quarter note. This is the most common variation type used by baroque composers and presents a good starting point for even more complex variation types, since there are enough music examples for learning and testing, and because it allows the representation of higher-scale elements in a rather straightforward way.
Figure 1 Structure of the system and process of composing a new melodic variation. A melody (previously harmonized by HARMONET) is passed to the supernet which predicts the current motif class $M_T$ from a local window. A similar procedure is performed on a lower time scale by the subnet which predicts the next motif note based on $M_T$ and the current harmony. The result is then returned to the supernet through the motif recognition component to be considered when computing the next motif class $M_{T+1}$.

3 A Multi-Scale Neural Network Model

The learning goal is twofold. On the one hand the results produced by the system should conform to musical rules. These can be melodic and harmonic constraints such as the correct resolving of dissonances or the appropriate use of successive interval leaps. On the other hand the system should be able to capture stylistic features from the learning examples, e.g. melodic shapes preferred by J. Pachelbel. The observation of musical rules and the aesthetic conformance to the learning set can be achieved by a multi-scale neural network model. The complexity of the learning task is reduced by decomposition in three subtasks (see Figure 1):

1. A melody variation is considered at a higher time scale as a sequence of melodic groups, so-called motifs. Each quarter note of the given melody is varied by one motif. These motifs are classified according to their similarity.

2. One neural network is used to learn the abstract sequence of motif classes. The question it solves is: What kind of motif 'fits' a melody note depending on melodic context and the motifs that have occurred previously? No concrete notes are fixed by this network. Since it works on a higher scale, it will be called supernet in the following.

3. Another neural network learns the implementation of abstract motif classes into concrete notes depending on a given harmonic context. It produces a sequence of sixteenth notes — four notes per motif — that result in a melodic variation of the given melody. Because it works one scale below the supernet it is called subnet.

The motivation of this separation into supernet and subnet arises from the following consideration: Having a neural network that learns sequences of sixteenth notes, it would be easier for this network to predict notes given a contour of each motif, i.e. a sequence of interval directions to be produced for each quarter note. Consider a human organist who improvises a melodic variation of a given melody in real time. Because he has to take his decisions in a fraction of a second, he must at least have some rough idea in mind about what kind of melodic variation should be applied to the next melody note in order to find a meaningful continuation of the variation. The validity of this concept was then confirmed by several experiments where motif classes, previously obtained from Pachelbel originals through classification, were presented to the subnet. After training the subnet was able to almost perfectly reproduce the Pachelbel originals. Hence the motif contour was shown to be one of the crucial elements characterizing melodic style. Therefore a neural network was introduced at a higher time scale. The training of this network really improved the overall behavior of the system and not just shifted the learning problem to another time scale.

4 Motif Recognition

In order to realize learning on different time scales as described above, we need a recognition component which is able to find a suitable classification of motifs. This can be achieved using unsupervised learning. We implemented a recursive clustering algorithm based on a distance measure which determines the similarity between small sequences of notes respectively intervals by comparing their motif contours. Figure 2 shows a motif and its corresponding representation. The result of hier-
architectural clustering is a dendrogram that allows comparison of classified elements on a distance scale. The algorithm is applied to all motifs contained in the training set. See [4] for a detailed account. One important problem is to find an appropriate number of classes for the given learning task. This will be discussed in section 6.

<table>
<thead>
<tr>
<th>1st interval</th>
<th>2nd interval</th>
<th>3rd interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>direction</td>
<td>size</td>
<td></td>
</tr>
<tr>
<td>↑</td>
<td>+1 0.2</td>
<td>-1 0.5</td>
</tr>
<tr>
<td>↓</td>
<td>+1 0.3</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2 Representation for motif classification and recognition

5 Interval Representation

In general one can distinguish two groups of motifs: Melodic motifs prefer small intervals, mainly primes and seconds, harmonic motifs prefer leaps and harmonizing notes (chord notes). Both motif groups heavily rely on harmonic information. In melodic motifs dissonances should be correctly resolved, in harmonic motifs notes must fit the given harmony. Small deviations may have a strong effect on the quality of musical results. Thus our idea was to integrate musical knowledge about interval and harmonic relationships into an appropriate interval representation. Each note is represented by its interval to the first motif note, the so-called reference note. This is an important element contributing to the success of our system. A similar idea for Jazz improvisation was followed in [1].

We have developed and tested various interval codings. The initial interval coding considered two important interval relationships: neighboring intervals are realized by overlapping bits, octave invariance is represented using a special octave bit. The activation of the overlapping bit was reduced from 1 to 0.5 in order to allow a better distinction of the intervals. This coding was then extended to capture harmonic properties as well. The idea was to represent in a similar way ascending and descending intervals leading to the same note. This is achieved using the complementary interval coding shown in the above table.

It uses 3 bits to distinguish the direction of the interval, one octave bit and 7 bits for the size of the interval. Complementary intervals such as ascending thirds and descending sixths have similar representations because they lead to the same note and can therefore be regarded as ‘harmonically equivalent’. A simple rhythmic element was then introduced using a tenuto bit (not shown in the above table) which is set when a note is tied to its predecessor. This final 3+1+7+1=12 bit interval coding gave the best results in our simulations.

Now we still need a representation for harmony. It can be encoded as a harmonic field which is a vector of chord notes of the diatonic scale. The tonic T in C major for example contains 3 chord notes — C, E and G — which correspond to the first, third and fifth degree of the C major scale (1010100). This representation may be further improved. We have already mentioned that each note is represented by the interval to the first motif note (reference note). We can now encode the harmonic field starting with the first motif note instead of the first degree of the scale. This is equivalent to rotating the bits of the harmonic field vector. An example is displayed in Figure 3. The harmony of the motif is the dominant D7, the first motif note is B which corresponds to the seventh degree of the C major scale. Therefore the harmonic field for harmony D7 (0101101) is rotated by one position to the right resulting in (1010110).

Starting with the first note B, the harmonic field indicates the intervals that lead to harmonizing notes B, D, F and G. In the right part of Figure 3 one can see the correspondence between bits activated in the harmonic field and bits set to 1 in the three interval codings. This kind of representation helps the neural network to directly establish a relationship between intervals and given harmony.

Figure 3 Example illustrating the relationship between complementary interval coding and rotated harmonic field. Each note is represented by its interval to the first note. The harmonic field indicates the intervals leading to harmonizing notes (i.e. B, D, F, G for harmony D↑7).
6 Performance

We carried out several simulations to evaluate the performance of the system. Many improvements could be found however by just listening to the improvisations produced by the system. One important problem was to find an appropriate number of classes for the given learning task. The following table lists the classification rate on the learning and validation set of the supernet and the subnet using 5, 12 and 20 motif classes. The learning set was automatically built from 12 Pachelbel chorale variations corresponding to 2220 patterns for the subnet and 555 for the supernet. The validation set includes 6 Pachelbel variations corresponding to 1396 patterns for the subnet and 349 for the supernet. The supernet and subnet were then trained independently with the RPROP learning algorithm [2].

<table>
<thead>
<tr>
<th># classes</th>
<th>supernet 5</th>
<th>12</th>
<th>20</th>
<th>subnet 5</th>
<th>12</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>learning set</td>
<td>91%</td>
<td>87%</td>
<td>88%</td>
<td>86%</td>
<td>94%</td>
<td>96%</td>
</tr>
<tr>
<td>validation set</td>
<td>50%</td>
<td>40%</td>
<td>38%</td>
<td>79%</td>
<td>83%</td>
<td>87%</td>
</tr>
</tbody>
</table>

The classification rate of both networks strongly depends on the number of classes, esp. on the validation set of the supernet. The smaller the number of classes, the better is the classification of the supernet because there are less alternatives to choose from.

We can also notice an opposite classification behavior of the subnet. The bigger the number of classes, the easier the subnet will be able to determine concrete motif notes for a given motif class. One can imagine that the optimal number of classes lies somewhere in the middle (=12 classes). This was then confirmed by comparing the results produced by different network versions.

We have also tested our neural organist on melodies that do not belong to the baroque era. Figure 4 shows a baroque-style harmonization and variation on the melody "Twinkle twinkle little star", used by Mozart in his famous piano variations. The result clearly exhibits global structure and is well-bound to the harmonic context.

7 Conclusions

We presented a neural organist improvising baroque-style variations on given melodies whose qualities are similar to those of an experienced human organist. The complex musical task is solved by cooperating neural networks together with unsupervised learning.

We believe that it is worth testing this multi-scale approach on learning examples of other epochs as well, e.g. on compositions of classical composers like Haydn and Mozart or on Jazz improvisations. Melodic variations of other epochs differ from those considered in this paper. We will therefore test if the system is able to reproduce style-specific elements of other kinds of melodic variations as well. Another interesting question is whether the global coherence of the musical results may be further improved adding another neural network working at a higher level of abstraction, e.g. at a phrase level (2-4 measures).

References


Figure 4  Melodic Variation on "Twinkle twinkle little star" invented by the Neural Organist