# A Supervised Learning Approach to Musical Style Recognition

Giuseppe Buzzanca

'N. Piccinni' State Conservatory of Music, Bari, Italy
and University of Bologna, Bologna, Italy
buzzanca@conservatoriopiccinni.it

**Abstract**. Musical style recognition is somehow intrinsic to human nature: the average layperson can tell the difference between a *motet* by Josquin and a symphony by Beethoven. But things change when it comes to computers. Musical style and its transformations take place in ways musicology is still unable perfectly to define. Up to now, computer-aided research on style has greatly privileged the rule-based approach (Baroni Dalmonte and Jacoboni, Lerdahl and Jackendoff, Ebcioglu, etc.). In this paper we use a supervised learning approach to create a "style recognizer" which is able to expose genre-attributable differences both within Palestrina's music (how is a mass different from a motet or a madrigal) and between Palestrina and other composers (either contemporary or younger/older). Our recognizer is implemented using a back-propagation neural network, with a number of innovative techniques to account for the special nature of musical input. Our technique makes no assumptions about the period or style of music being studied, but our experiments have focused on Palestrina. Results are promising: our technique seems to possess those generalization capabilities which are essential when dealing with a matter of such proteiform complexity as musical style.

## 1. Introduction

It may seem that if we were able to define the syntax of a language with precision and rigor, then we would fully comprehend its style and structure.

Influenced by Chomsky's ideas, a number of scholars developed models of musical grammars. To cite just a few examples, the popular GTTM (Generative Theory of Tonal Music) by Lerdahl and Jackendoff (1983):[1] combining Chomskian linguistics with Schenkerian analysis, the authors describe how a listener, who is expected to be familiar with tonal music, instinctively comes to a complete intelligence of musical structure. Such a grammar includes a system of rules that assigns to every piece of music its own analysis.

[1] Even though the authors say that their theory is not a computational one since "achieving computability [...] requires a much better understanding [...] than exists at present" with regard to many musical and psychological issues (Lerdahl and Jackendoff 1983, p. 55).

Kemal Ebcioglu (1988) has realized an expert system (CHORAL) that can compose chorales in the style of Bach, by harmonizing a given soprano part. CHORAL uses a fairly complex knowledge base (approximately 400 rules) concerning both analytical and harmonic aspects.

Baroni's research group, have investigated different melodic repertoires in order to detect the particular rules behind each one of them as well as the general rules behind all of them. Their research has firstly concerned Bach's chorales (Baroni and Jacoboni, 1978) and, more recently, the *arie da camera* by the Italian composer Giovanni Legrenzi (Baroni, Dalmonte and Jacoboni, 1999). In the latter case, particular attention was paid to connections with the harmony and the poetic text. Their principal hypothesis is that a melody in the style of Legrenzi can be composed starting from a kernel consisting of two notes which will successively generate a melody according to the rules of a given grammar.

Buzzanca (Buzzanca, 1999) has realized an expert system which uses a knowledge base derived from Baroni Dalmonte and Jacoboni's rules (Baroni, Dalmonte and Jacoboni, 1999) to see 1) whether or not these rules successfully account for the structure of Legrenzi's works, and to formulate hypotheses about the circumstances in which the rules are broken; 2) to assess (Buzzanca, 2001) the adequacy-inadequacy of rules in describing a different repertoire (although historically and stylistically related) from the one they were created to model.

Nevertheless, the use of grammars to the modeling of musical style is unsatifying for several reasons: Camilleri (1992) has pointed out that they are vulnerable with regard to the creation of *ad hoc* rules as well as for their incapacity to represent ambiguity in musical cognitive processes. Ovans and Davison (1992) have demonstrated that the satisfaction of constraints alone is not sufficient for musical results.

We believe that if we attempt to capture musical style through grammatical descriptions, we end up trying to construct grammars that become so unwieldy that they cease to serve the generalizing functions that are the definition and purpose of grammars in the first place. The fundamental problem is that it is extremely difficult, if not impossible, to construct a grammar that will include all true instances of a language, and exclude all false ones. As a result, only a small fraction of the examples produced by a grammar are actual instances of the style it is supposed to model.

Our approach is to use machine learning to develop a musical style recognizer. Recognition can be viewed naturally as a supervised learning task. The learner is given examples of music in the specified style, together with non-examples, and learns from them to distinguish between examples and non-examples. However, musical style recognition is different from traditional supervised learning tasks due to the free-form nature of the input, and the depth of structure. Considerable care is required in the design of the learning model.

We have chosen back-propagation neural networks for our learning approach for a number of reasons. They have an excellent track record in complex recognition tasks in problems such as face recognition (see Lawrence *et al.* 1996). They are capable of inducing the hidden features of a domain, that may be elusive to a rule-based approach. Their flexible design allows us to encode musical structure in the model. They have been successful in other musical tasks, such as composition (Mozer 1991) and improvisation of *baroque-style* melodic variations (Hörnel and Degenhardt, 1997).

We do not view learning-based style recognition as a replacement for grammar-based approaches, but as complementary. Our style recognizer is incapable of generating examples of a style, only of recognizing which candidate examples are legitimate. A grammar model, on the other hand, is capable of generating examples, but also generates many non-examples. The two naturally go hand–in-hand. If one has a good generator and recognizer, one can use the generator to generate many candidate examples, and use the recognizer to filter out the bad ones. Furthermore, one can use a recognizer to provide an objective measure of the quality of a generator: how well is the recognizer capable of distinguishing generated examples of a style from actual examples?

## 2. Neural networks and music research

A quantitatively notable collection of studies dealing with the development and use of connectionist models for composition is available in literature: Todd (1989, 1991), Lewis (1991), Mozer (1991, 1994), Stevens and Wiles (1993), Bellgard and Tsang (1994) and, more recently, Hörnel and Ragg (1996), Hörnel and Degenhardt (1997), Hörnel (1998) Hörnel and Menzel, (1999) and Höthker, Hörnel and Anagnostopoulou (2001). However, fewer are the examples of application of neural networks in the field of musical style recognition; almost all of them seem unsatisfactory since they deal with too few and/or partial aspects of musical style. So in Bellgard and Tsang (1996), where the Effective Boltzmann Machine (an extension of the Boltzmann Machine neural network) is used to distinguish between the same fragment of melody which has been given two different styles of harmonization (one harmonization in baroque style, the other in the style of the norwegian chorale) and Feulner (1993) where the classification of the pieces is operated solely with respect to their harmonization.

It is needless to say though, that one can't reduce musical style to a single *genre* (the choral) and adress all concerns solely toward harmony; musical style is a vast, general phenomenon, which could rather be dealt with by means of extensive comparative analyses tending to define global processes of style transformation. In other words, both papers wish to deal with the double aim of analyzing and producing music but at the very end seem to concentrate mainly on the purpose of learning harmonies for harmonization/composition tasks: but in so doing the idea of realizing a true, general purpose stylistic recognizer fades away.

On top of that, test materials in the above-mentioned research appear to be too narrow to reassure that a real generalization process has taken place: training set consisted of only five baroque chorales and three Norwegian chorales in Bellgard and Tsang (1996) and twenty chorales in Feulner (1993).

# 3. Problem description

## 3.1 Repertoire

As we stated before, we believe that musical style is a vast, general phenomenon, which could rather be dealt with by means of extensive comparative analyses tending to define/quantify global processes of style transformation.

For this reason we have chosen to have in our sample every degree of difficulty with regard to stylistic recognition; at the same time, we want to underline that even though we had to pick up a list of specific composers, a similar approach could have been equally possible considering other composers belonging to other periods without weakening its theoretical range and importance. What we want is indeed to come to a stylistic recognizer that is the most general possible: the analyzed repertoire has to be considered as one of the numerous (virtually infinite) 'opportunities' to experiment with in this challenging field.

**Table 1.** List of composers.

| Composer | Birth/Death |
|---|---|
| 1. Josquin Des Prez | 1440-1495 |
| 2. Andrea Gabrieli | 1510-1586 |
| 3. Giovanni P. da Palestrina | 1525-1594 |
| 4. Orlando di Lasso | 1532-1594 |
| 5. Thomas L. Da Victoria | 1548-1611 |
| 6. Jacobus Handl | 1550-1591 |
| 7. Luca Marenzio | 1553-1599 |
| 8. Giovanni Gabrieli | 1554-1612 |
| 9. Gastoldi Gian Giacomo | 1555-1609 |
| 10. Carlo Gesualdo da Venosa | 1560-1613 |
| 11. Hans Leo Hässler | 1564-1612 |
| 12. Claudio Monteverdi | 1564-1643 |
| 13. Alessandro Scarlatti | 1660-1725 |

The composers we chose (see above tab. 1) constitute a thick group of sublime Renaissance contrapunctists, with the exception of Josquin and Scarlatti. The question is: how well will our recognizer do with the large number of nuances it will have to deal with? In other words: Josquin is the one who completes the grand Flemish creative cycle. So, even though we have to think of him as chronologically earlier and stylistically different in comparison to composers like Marenzio or Monteverdi, we will also have to consider that he probably constituted an archetype for the composers of the following generation (among whom we may include all the others except Scarlatti). For this reason, to succeed in perceiving differences between his music and music by composers of the Renaissance doesn't seem an entirely trivial assignment.

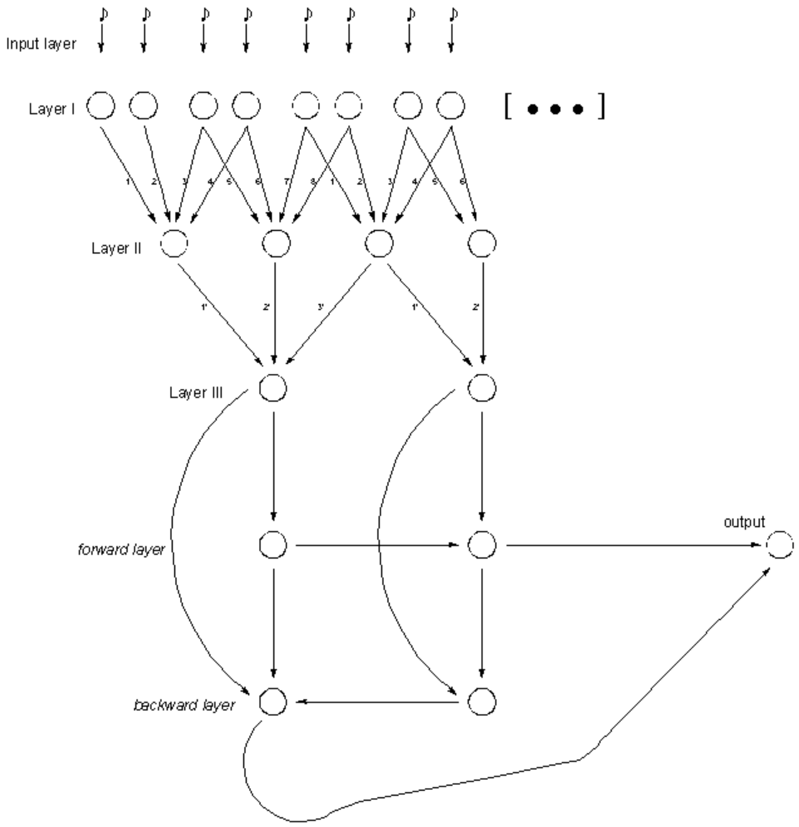
As we said, we have different degrees of difficulty: how much does the contrapuntal technique differ in Gastoldi, who writes his *balletti* in a ‘light’ form (since the *balletto* is conceived in the style of dance) but is however parallel to the madrigal in comparison to a *kyrie* by Palestrina? And so forth with all possible combinations: if we concede that a balletto by Gastoldi might differ in wide measure from a *kyrie* by Palestrina, it is not obvious anymore that we can catch the slight differences between the sacred music composed by Palestrina and the one composed by T. Da Victoria, in whose style incredible importance had the deep study and assimilation of Palestrina’s style. And here again, we might expect to find a trace that an opera writer like A. Scarlatti (used to the elaboration of complex *fioriture* and vocal virtuosities for the star-singers of the time) composes church music that greatly differs from Da Victoria’s one.

Summarizing: we had to choose a particular repertoire even though nothing forbids us from extending the scope of this research to other repertoires. This particular repertoire constituted one of the many opportunities to probe the effectiveness of our approach with regard to the analysis of musical style: 1) even though it might be considered homogeneous (since it consists for the most part of sacred/polyphonic music), it still has instances of various genres (mass, madrigal, motet) and several idiolects are represented (the list of composers covers almost three hundred years of the history of music); 2) the test set is wide enough to reassure us that a real process of generalization has taken place; 3) we took into consideration several determining factors, overcoming the limit that consists of endowing the network of an input layer having a fixed number of units, which isomorphically match fixed-length fragments of musical phrases (see Linster, 1989). This last type of segmentation constitutes a simplification in comparison to real life, since the ‘amputation’ of musical stimuli into sequences of predetermined length is not realistic neither at a perceptive nor at a theoretical/compositional level (see Sloboda, 1985; Imberty, 1987; Smoliar, 1992). Thus our approach avoids the serious loss of sense which would result from feeding the network with data separated from their musical context; 4) several peculiarities of music were taken into consideration, by encoding the data in a special way both at the micro level (melody, rhythm, melodic countour, etc.) and at the macro level (‘directionality’ of music, regarding the highest level of hierarchy, and feeding it to a directed network, etc.)

### 3.2 The Neural Network Model

Our neural network model is designed with three purposes in mind: to account for musical input of varying length in an even manner; to model the hierarchical nature of musical recognition; and to capture the importance of directional flow in music. A schema of our network is shown in Figure 1 below.

**Fig. 1.** Schema of the network.



At the heart of our method is the technique of *shared weights* developed by LeCun *et al.* (1989). The idea is quite simple. In an ordinary neural network, there is a separate learnable weight associated with every edge between two nodes. Using shared weights, many of these weights are forced to be the same, in a symmetric way. Thus, all edges belong to a certain type, and weights are associated with types of edge rather than individual edges. An example of a type of edge in our network is the edge from the node representing the interval from the second to the third beat in a measure, to a node summarizing the entire measure. This edge will have the same weight for all measures. However, the edge from the node represent the interval between the first and second beats to the node for the whole measure will be different. In Figure 1, edges with shared weights have the same label.

Shared weights have several advantages. As seen in the above example, they capture the fact that different edges represent the same concept, and so the learning is shared between all those edges. Furthermore, it is well known that the sample complexity of a neural network, and therefore the number of training examples needed to learn accurately, is closely related to the concept of *VC-dimension* (see Vapnik, Levin and LeCun, 1994), which depends on the number of different weights.

Shared weights greatly reduces the total number of different weights, and therefore the amount of training data needed. This is important, because a complex network structure is required to represent the subtleties of musical style, but there is not enough data to learn an ordinary network of sufficient complexity.

The shared weights technique also allows us to deal with the issue of variable input size. Simply put, while the number of edges is allowed to vary with the size of the input, the number of edge types, and therefore the number of phrases, is constant. Given a musical phrase of a certain length, we simply grow a network of sufficient size to represent the entire phrase. The same thing will be learned for phrases of all different types, namely, the weights associated with each of the edge types. Thus the ellipsis in the diagram can be filled with any number of measures, all with similar structure.

To get at the multi-level analysis, we use multiple hidden layers, each summarizing the previous level. The first hidden layer summarizes the information in a single beat. The second layer takes the output of the first layer and summarizes two beats. Note that the second layer actually has four inputs. That is because in the first layer we have two patterns: on-beat to off-beat and off-beat to on-beat. Similarly, at the second layer we have strong-weak patterns and weak-strong patterns. The theory is that a particular style might be characterized by what happens in either style. The hierarchy continues in a similar way: the third level summarizes four beats, and so on. In our experiments, we limit the network to three levels.

To get at the directionality of music, we took the output of the highest level of hierarchy and fed it to a directed network. The directed network consists of a sequence of nodes, from the beginning of the music till the end, where each node depends on the previous node in the sequence and on the corresponding top-level node in the hierarchy. This subnetwork is shown as the “forward layer” in the figure. A danger with this design is that the end of the music ends up having a much more direct influence on the outcome than the beginning. To compensate for this, we introduce a similar “backward layer” that captures the “backward flow” of the music. Between them, these two layers model the directional impulse.

The diagram in Figure 1 is only a schema of our actual network. Each node in the diagram actually represents a set of nodes. Hidden layers typically use between two and six nodes at each point; for example, there may be four nodes representing the third-level strong-weak summary. Each such node has an input from all the relevant second-level hidden nodes.

The input consists of four nodes for each unit of music. The unit chosen was half a beat, i.e. an eighth note for music in 4/4 time. This is the smallest unit that occurs frequently in Palestrina, but our method adapts easily to smaller units. Although sixteenth notes do occur occasionally in Palestrina, we believe that building a network for quarter beat analysis would increase the complexity too much to justify modeling these occasional events accurately. As it turns out, we can deal with sixteenth notes quite well, as described below.

Out of the four input units for each half-beat, two encode rhythmic information, and two encode pitch information. The rhythm encoding is a generalization of that of Carpinteiro (1995). The first unit encodes the onset of a note. Its value is 0 if no note begins at that time, otherwise the value is the logarithm of the duration in half-beats, e.g. 1 for an eighth note, 2 for a quarter note, 3 for a half note and so on. This representation allows us to encode duration, which we believe to be important in style identification, but in such a way as to minimize the impact of very long notes. The second rhythm unit corresponds to whether a note is sounded at a point. It is 1 for sounded notes and 0 for rests.

Pitch is encoded with two nodes. The first encodes the absolute pitch in semitones, relative to the first note of the melody. The second is the relative pitch to the previous sounded note. While there is redundancy here, as either one could be deduced from the other, both are useful for style recognition. The first expresses aspects such as range and contour, while the second provides intervallic information. Indeed experience shows that representing the same data in multiple ways can be useful to neural network classifiers.

The double encoding of pitch also allows us to deal with sixteenth notes. The first of the two sixteenth notes is usually the more important, so we use it to encode the pitch at the point of the sixteenth notes. However, for the interval input of the next note, we take the second of the two sixteenth notes as the previous note, since that is the one defining the interval. In this way we are able to capture key aspects of both notes, without having to greatly grow the network to accommodate a sixteenth-note scale.

An example of the encoding of the network is shown below.

```
(* weights are pointers to floats to allow weight
sharing *)

type weight = float ref

(**************Comments on the network**************)
(*                                                 *)
(* null layer: contains the single node0           *)
(* layer 0: input layer                            *)
(* layer 1: nodes from layer 0 connected pairwise  *)
(* layer 2: four nodes from layer 1 connected      *)
(* layer 3: three nodes from layer 2 connected     *)
(* forward layer : actually, not a layer, but a    *)
(*                 sequence of single nodes        *)
(*                 connected in forward direction, *)
(*                 assuming there are n nodes in   *)
(*                 layer 3, zero-1-2-3-....-n-out  *)
(* backward layer: again, not a layer, but a       *)
(*                 sequence of single nodes        *)
(*                 connected in backward direction,*)
(*                 assuming there are n nodes in   *)
(*                 layer 3, zero-n-(n-1)-.. -1-out *)
(* output layer:   contains the single node "out"  *)
(*                                                 *)
(***************************************************)
```

```
(* holds the values for all the different shared
weights in the network *)
type net_weights = {
  null_to_one   :     weight;
  null_to_two   :     weight;
  null_to_three :     weight;
  null_to_forward :   weight;
  null_to_backward :  weight;
  null_to_out :       weight;
  zero_to_one_1 :     weight;
  zero_to_one_2 :     weight;
  one_to_two_1 :      weight;
  one_to_two_2 :      weight;
  one_to_two_3 :      weight;
  one_to_two_4 :      weight;
  two_to_three_1 :    weight;
  two_to_three_2 :    weight;
  two_to_three_3 :    weight;
  three_to_forward :  weight;
  three_to_backward : weight;
  within_forward :    weight;
  within_backward :   weight;
  forward_to_out :    weight;
  backward_to_out :   weight;
}


(* a node has a list of nodes its connected to and the
weights for the links *)
type node = (int * weight) list

(* neural net lists the nodes in the network in
topological order *)
type neuralnet = node array


exception Short_Phrase

let generate_test_weights () =
  {
    null_to_one       = ref 0.01;
    null_to_two       = ref 0.02;
    null_to_three     = ref 0.03;
    null_to_forward   = ref 0.04;
    null_to_backward  = ref 0.05;
    null_to_out       = ref 0.06;
    zero_to_one_1     = ref 0.11;
```

```
      zero_to_one_2      = ref 0.12;
      one_to_two_1       = ref 1.21;
      one_to_two_2       = ref 1.22;
      one_to_two_3       = ref 1.23;
      one_to_two_4       = ref 1.24;
      two_to_three_1     = ref 2.31;
      two_to_three_2     = ref 2.32;
      two_to_three_3     = ref 2.33;
      three_to_forward   = ref 3.4;
      three_to_backward  = ref 3.5;
      within_forward     = ref 4.4;
      within_backward    = ref 5.5;
      forward_to_out     = ref 4.6;
      backward_to_out    = ref 5.6
    }
```

## 4. Experiments and results

We run several simulations to test our model. The learning set (consisting of thousand of patterns) was automatically built extracting the melodic phrases belonging to around 400 pieces of music (madrigals, motets and masses by the composers mentioned in table 1).

The learning process was time-consuming: it took around 20 hours for the network to go through all the generated patterns.

After learning, the network could recognize the training examples correctly, i.e. it could discern Palestrina’s phrases from other composers’ phrases. But interestingly enough, with the test set (that the network had not seen before), results were also good (average 97,09%): this probably means that our model is able to catch features which are important in order to operate a distinction.

## 5. Conclusions

We believe that the results obtained show that neural networks are able to deal with some challenging problems pertaining to stylistic recognition.

Even though this is just a first and partial effort toward a better understanding of musical style, neural networks seem particularly suitable in a field where it appears to be quite difficult (if not impossible) to elaborate symbolic descriptions.

## References

1. Baroni, M., Jacoboni, C.: Proposal for a grammar of melody. The Bach chorales. Presses Universitaire de Montréal, Montréal (1978).
2. Baroni, M., Dalmonte, R., Jacoboni, C.: Le regole della musica. Indagine sui meccanismi della comunicazione. EDT, Torino (1999).
3. Bellgard, M. I., Tsang, C. P.: Harmonizing music the Boltzmann way. Connection Science, 6, (1994), 281-298.
4. Bellgard, M. I., Tsang, C. P.: On the Use of an Effective Boltzmann Machine for Musical Style Recognition and Harmonization. In: Proceedings of the 1996 International Computer Music Conference. International Computer Music Association, San Francisco (1996).
5. Buzzanca, G.: Analisi di strutture melodiche al calcolatore. University of Bologna, Bologna (1999).
6. Buzzanca, G.: Melody Parser: un sistema esperto per il riconoscimento stilistico. In: Proceedings of the Symposium ‘Capire e creare la musica. Algoritmi geometrici e frattali, modelli fisici e strutture percettive’. Seconda Università degli Studi di Napoli, Napoli (2001).
7. Camilleri, L.: Computational Theories of Music. In: Computer Representation and Models in Music. Academic Press, San Diego (1992) 175 - 177.
8. Carpinteiro, O.A.S.: A neural model to segment musical pieces. In: Proceedings of the II Brazilian Symposium on Computer Music. Canela (1995) 114-120.
9. Dannenberg, R., Thom, B., Watson, D.: A machine learning approach to musical style recognition. In: Proceedings of the 1997 International Computer Music Conference. International Computer Music Association, San Francisco (1997).
10. Ebcioglu, K.: An Expert System for Harmonizing Four-Part Chorales, Computer Music Journal 12 (1988).
11. Feulner, J.: Neural Networks that Learn and Reproduce Various Styles of Harmonization. In: Proceedings of the 1993 International Computer Music Conference. International Computer Music Association, San Francisco (1993).
12. Hörnel, D., Ragg, T.: Learning Musical Structure and Style by Recognition, Prediction and Evolution. In: Proceedings of the 1996 International Computer Music Conference. International Computer Music Association, San Francisco (1996).
13. Hörnel, D., Degenhardt, P.: A Neural Organist inprovising baroque-style melodic variations. In: Proceedings of the 1997 International Computer Music Conference. International Computer Music Association, San Francisco (1997).
14. Hörnel, D.: A Multi-scale Neural-Network Model for Learning and Reproducing Chorale Variations. Computing in Musicology, 11 (1998).
15. Hörnel, D., Menzel, W.: Learning Musical Structure and Style with Neural Networks. Computer Music Journal, 22 (1999) 44-62.
16. Höthker, K., Hörnel, D., Anagnostopoulou, C., Investigating the Influence of Representations and Algorithms in Music Classification. Computers and the Humanities 35 (2001) 65–79.
17. Lawrence, S., Giles, C. L., Chung Tsoi, A., Back, A. D.: Face Recognition: A Hybrid Neural Network Approach. Technical Report, UMIACS-TR-96-16 and CS-TR-3608. Institute for Advanced Computer Studies University of Maryland, College Park (1996).
18. LeCun, Y., Generalization and Network Design Strategies. In: Pfeifer, R., Schreter, Z., Fogelman, F., Steels, (eds.): Connectionism in Perspective. Zurich (1989).
19. LeCun, Y., Bengio, Y.: Convolutional networks for images. Speech, and time-series. In: Arbib, M. A., (editor): The Handbook of Brain Theory and Neural Networks. The MIT Press, Cambridge (1995).

20.Lerdahl, F., Jackendoff, R.: A Generative Theory of Tonal Music. The MIT Press, Cambridge (1983).

21.Lewis, J. P.: Creation by refinement and the problem of algorithmic music composition. In: P. M. Todd & D. G. Loy (eds.), Music and Connectionism. MIT Press, Cambridge (1991) 212-228.

22.Linster, C.: Get Rhythm, a Musical Application for Neural Network. Arbeitspapiere der GMD (1989).

23.Mozer, M. C.: Connectionist music composition based on melodic, stylistic and psychophysical constraints. In: Todd, P. M., Loy, D. G. (eds.): Music and Conneclionism. MIT Press, Cambridge (1991) 195-212.

24.Mozer, M. C.: Neural network music composition by prediction: Exploring the benefits of psychoacoustic constraints and multi-scale processing. Connection Science 6 (1994) 247-280.

25.Ovans, R., Davison, R.: An Interactive Constraint-Based Expert Assistant for Music Composition. In: Proceedings of the Ninth Canadian Conference on Artificial Intelligence. Vancouver (1992), 76-81.

26.Sloboda, J.: The musical mind: The Cognitive Psychology of Music. Clarendon Press, Oxford (1985).

27.Smoliar, S.: Representing listening behavior: Problems and prospects. In: Balaban, M., Ebcioglu, K., Laske, O. (eds.): Understanding music with AI. MIT Press, Cambridge (1992).

28.Stevens, C., Wiles, J.: Representations of tonal music: a case study in the development of temporal relationships. In: Mozer, M. C., Smolensky, P., Touretzky, D. S., Elman, J. E., Weigend, A. S., (eds.): Proceedings of the 1993 Connectionist Models Summer School. Eribaum Associates, Hillsdale (1993) 228-235.

29.Todd, P. M. A connectionistic approach to algorithmic composition. Computer Music Journal 13 (1989) 27-43.

30.Todd, P. M.: Addendum to: A connectionist approach to algorithmic composition. In: Todd, P. M., Loy, D. G., (eds.): Music and Connectionism. MIT Press, Cambridge (1991) 190-194.

31.Vapnik, V., Levin, E., LeCun, Y.: Measuring the VC-dimension of a Learning Machine. Neural Computation 6 (1994) 851-876.