

Neural Networks for Applications in the Arts

Peter M. Todd

Department of Psychology
Jordan Hall, Building 420
Stanford University
Stanford, CA 94305
todd@psych.stanford.edu

NOTE: \fIword\fP means *word* and \fBword\fP means **word**

Introduction

Traditionally, computer applications in the arts have been pretty much like computer applications in any other field: you'd like the computer to create a certain sort of output, so you construct an algorithm for it to follow step by step which will result in that output. This is a fine approach when you know what you're going for and where the production process should be headed at each step, that is, when you can precisely specify the rules involved in generating the output. Programming a computer to invert a matrix or control an assembly-line drill-punch is certainly like this, and the production of a variety of types of art are amenable to this approach as well. But there are also a wide range of instances in which we as artists may not want to, or be able to, specify a set of rules by which our next computer artwork is to be created. Sometimes it may just be too difficult or time-intensive to list the intricacies of a particular creative process in a fashion definite enough for a computer to follow; and other times, we may have no idea how to even go about constructing an appropriate set of rules (what are the "rules" to follow in creating a Rodin sculpture?). At times like these, a new approach is needed.

A new approach to computer art applications is provided by the rapidly expanding field of neural networks. Rather than operate in the traditional style of preprogrammed rule-following systems, neural networks have the power to \fIlearn\fP to produce specific types of outputs from specific inputs, based on examples they are taught. Thus, instead of having to specify \fIhow\fP to create a certain artwork, the artist can instead teach a network \fIexamples\fP of the desired output, and have the network generate new instances in that style. We need not specify the steps involved in creating a Rodin sculpture--we can just collect instances of the sorts of sculpture we'd like to get and use those to train a network. This sort of thing is the promise of neural network applications in the arts, stated in a very provocative and exaggerated manner. But we are beginning to realize this promise in limited domains, and the space for further applications and explorations of these techniques is wide open.

In this paper, I will briefly describe neural networks, some of what they can and can't do, some of what they have done already in the arts, primarily music, and some of what they can be applied to in other areas. I will also give a few pointers to the literature for further reading and exploration of these topics; I hope people will be encouraged to pursue all of these avenues and see where their creativity leads them.

Neural Networks

The standard von Neumann-style computer programming paradigm is based on the concept of a single powerful central processing unit that performs a sequence of operations on the contents of an external memory. The neural network computing paradigm, in contrast, is based largely on the notion of "brain-style computation" (Rumelhart, 1990), in which a large number of very simple processing units act simultaneously on a distributed pattern of data--hence the common name, parallel distributed processing (Rumelhart and McClelland, 1986). The analogy here is to the type of computation done by the vast numbers of simple neurons in the brains of living creatures.

There is a vast assortment of mathematical formulations of different neural network types, but for our purposes we can think about a simple case in which a network consists just of a set of simple processing units, connected by a set of weighted links. The currency of the network is numerical values--the units all take in numerical values passed to them on the links from other units, and they in turn produce a single numeric output value with they pass on their output links to still other units for further processing. A subset of the units will typically have a special status as input units, which take some pattern of "information" (represented as a set of numerical values) from the external world into the network for processing. Similarly, there is a set of output units, whose final processed values are taken as the end product of the network's computation. Along the path of links inbetween these two sets can be a collection of "hidden" units, which perform the network's main work. They transform the input values into successively more "useful" representations on the way to the final output.

All of the units in a network (of the sort we're describing here) do exactly the same thing, summing up their inputs from other units, and putting that value through a "squashing function" which maps the unit's output into a restricted range in a non-linear fashion. This non-linearity allows the network to make distinct categorizations and judgments about its input. Since all of the units act the same, the only way to get two networks with the same unit setup to act differently is to have different weights on their links between the units. And the easiest way to set the weights on those links is to use a learning method.

Learning is a key feature of neural networks, and a big advantage they have over other computational frameworks. Learning comes naturally in networks, but is typically ad-hoc, if possible at all, in rule-based systems. The weights in a network are typically learned by a type of method known as *gradient descent*, in which small adjustments are made to appropriate (mathematically determined) weights to lessen the network's performance error. This error is a measure of how far the network's actual current performance deviates from its desired performance--that is, how far the outputs the network is now creating differ from the target outputs in the examples the network is trying to learn. For instance, if the example we're training the network on is to sculpt a bust, and its output corresponds to a sculpted automobile, there would be a large error difference there, and a lot of weights would need to be changed in the network. If its output corresponds instead to a head without eyes, the error, and the necessary weight-changed, would be correspondingly smaller. Tiny adjustments are made to the weights during each step in training the network, and then the example is tried again, to see how far the target from the current output is now; and then again the weights are changed slightly. After many cycles of this test-and-

adjust, the network's performance will hopefully have come to match the desired examples closely enough, and training can stop. After this, the network can be used to create new outputs.

(For more details on the actual implementation and mathematics of all this, presented from the standpoint of musical applications, see Dolson, 1991; Rumelhart, Hinton, and Williams, 1986, provide the original complete derivation of a particular popular learning method known as "back propagation of error"; McClelland and Rumelhart, 1988, provide a tutorial introduction to a variety of neural network approaches and the theory underlying them, and even provide software on diskettes for running your own simulations on Macs and PC's; and Trubitt and Todd, 1991, present these issues in a little more detail but at a very informal level, again for application to music.)

Because of the "massive parallelism" achieved by many units operating simultaneously on a distributed pattern of information, small changes in the network will affect its performance rather little. Removing one unit might not do too much to the network's performance error, because several other units also take part in the calculations it performed and can "fill in" for it, so to speak. In contrast, removing a single line from a standard algorithm can render the entire thing useless--catastrophic failure. A further consequence of this parallel distributed processing is that small changes in the inputs will result in small changes in the output. This means the network will generalize to new inputs in reasonable ways--if it sees something new but not too different from a previously-seen input pattern, it will output something similar to the associated old output pattern. So if for instance an input of values "1,0,0,1,0,1,1" resulted in an output of a sculpted ram's head from a specifically trained network, an input of values "1,0,0,1,0,1,0" instead might result in a similar output, but with slightly shorter horns. This generalization to new inputs in sensible ways is the basis of many of the creative applications of neural networks.

So far most of the examples I've used for network art applications have centered on the production of new output, with little regard for the input end of things. But one of the tasks neural networks are best suited for is more standard input-output processing, taking a certain input set of values and transforming it in particular ways to generate the (more-or-less related) output. More specifically, the inputs and outputs can represent signals of some sort, whether auditory, visual, or otherwise, and the network can perform some kind of signal processing function, removing noise, rotating an image, filling in colors, etc. Networks can also be used in interesting dynamic ways, for instance by connecting their outputs back to their inputs to create a feedback loop that generates a sequence of outputs. As nonlinear dynamic systems of this type, sequential networks can have attractors and cycles that are useful in a variety of contexts (see Todd, 1991a, for a musical application, and Jordan, 1986, for guiding arm movement). And as adaptive systems that learn to respond in appropriate ways in changing situations and environments, neural networks have found extensive use in the control of motion and navigation in robot and other systems. These are the sorts of tasks networks are particularly suited for, and as we will see, they can be particularly useful in a lot of artistic applications.

There are, though, still several things which the neural network approach is not good for at present. Networks are very good at learning the low-level details needed to perform a certain input-output mapping, for instance in learning to produce the next measure of music as output, given the previous measure as input. They are notoriously poor, however, at picking up the higher-level structure that organizes the lower-level features, again for instance missing the global movement

away from and back to the tonic in the succession of measures in a piece of music. In fact, the higher-level the task in general, the fewer successful applications there have been of neural networks, where high-level here means things such as language processing as opposed to speech, image understanding as opposed to processing, motion planning as opposed to production. And focussing as they do on \fIsub-\fPsymbolic processing, distributing meaning across a number of units and connections, neural networks have not made much inroad as yet into the traditional symbol-processing arenas of symbolic reasoning, syntax and semantics, expert systems, and rule-following. But then again, as I described in the introduction, the beauty of the parallel distributed processing paradigm is that its strengths lie elsewhere.

Applications in the Arts

We now turn to the artistic uses of neural networks. Currently, this is a wide-open field; exploration has just begun in most cases, and we've barely scratched the surface of possibilities. The ideas below are mostly speculations on what networks \fIcould\ fP do, the sorts of tasks they \fIcould\ fP be applied to in the arts, sometimes based on applications that have already been done in scientific or engineering domains, and sometimes just based on imaginative speculation. As such, these ideas are intended to spark people's imaginations further in the search for innovative uses of this powerful and flexible new technology.

The main place where neural networks have been put to creative and artistic use so far is in music, as witnessed by the recent publication of the book, \fIMusic and Connectionism\ fP (Todd and Loy, 1991). Several applications have been done in this area, ranging from psychological models of human pitch, chord, and melody perception, to networks for algorithmic composition and performance control. Generally speaking, the applications here (and in other fields) can be divided into two classes: "input" and "output". The input side includes networks for recognition and understanding of a provided stimulus, for instance speech recognition, or modelling how humans listen to and process a melody. Such applications are useful for communication from human to machine, and for artistic analysis (e.g. musicologically, historically) of a set of inputs. The output side includes the production of novel works, applications such as music composition or drawing generation. "Input" tasks tend to be much more difficult than "output" tasks (compare the state-of-the-art in speech recognition versus speech production by computers), so most of the network applications so far have focussed on creation and generation of output, but continuing research has begun to address this imbalance.

On the "input" side in musical applications, Sano and Jenkins (1991) have modelled human pitch perception; Bharucha (1991) (and others) have modelled the perception and processing of harmony and chords; Gjerdingen (1991) has explored networks that understand more complex musical patterns; and Desain and Honing (1991) have devised a network for looking at the quantization of musical time and rhythm. Dolson (1991) has also suggested some approaches to musical signal processing by neural networks, including instrument recognition, generation, and modification. In this regard, musical applications of networks have much to gain from the vast literature on networks for speech processing (primarily recognition--see Lippmann, 1989).

On the "output" side, several network models of music composition have been devised. Todd (1991a) and Mozer (1991) use essentially the dynamic sequential network approach mentioned earlier, in which a network is trained to map from one time-chunk of a piece of music to the following time-chunk (e.g. measure N as input should produce measure N+1 as output). The network's outputs are then connected back to its inputs for the creation phase, and a new measure 1 is provided to begin the network down a new dynamic path, creating one measure after another, and all the while incorporating the sorts of features it learned from its training examples. In this way, new pieces that have a sound like Bach or Joplin (or a combination of both!) can be created, if the network is first trained on these composers. But the problems mentioned earlier of lack of higher-level structure emerge, and these compositions tend to wander, having no clear direction, and seldom ending up anywhere in particular. Approaches for learning and using hierarchical structure are being devised, and Lewis (1991a) describes one such method, in which the inputs to a network, rather than the weights in the network, are modified during a learning stage, to produce an input which has a specified form or character. Kohonen et al. (1991) present still another method of composition, which uses a network-style approach to build up a context-free grammar that models the music examples it's trained on.

Networks can also be used to generate musical performance parameters and instructions, as Sayegh (1991) demonstrates in his paper on a network method for choosing correct chord fingering for a simple melody. Many other musical performance applications are possible, from synthesizer control to automatic rhythmic accompaniment generators; Todd (1991b) discusses some of these possibilities along with further ideas for musical applications of neural networks.

Neural networks have been applied to a variety of image-processing tasks, from video image compression to aspects of computer vision such as image segmentation and object recognition. As signal processors, networks should find wide application in image enhancement, color adjusting or altering, edge and line modification, texture processing, etc., all based on learned mappings from input pictures to desired outputs. As another example, Chen et al. (1990) have developed a network for choosing a palette of colors (for use on a Macintosh screen) from a small input set of preferences.

At a higher level, networks can be used to generate actual drawings. The tricky part here is figuring out how to encode the drawings into a numerical form that the network can work with; Lewis (1991b) uses spline coefficients to encode simple drawings that are learned by a network similar to the one he has used for music (1991a). While the results are so far admittedly rather crude, they point in a direction of great promise. Writing, calligraphy, and the creation of new fonts is another area networks can profitably tackle. Grebert et al. (1991) have developed a system which will generate a complete font in a certain style, given just a few letters in that font to begin with. Again often the network's output is questionable, but it does an interesting job of generalization, to say the least, and enhancements of this technique could yield a bonanza of new letter and writing styles. (The area of neural network handwriting recognition is also being widely explored, but again there as for speech, the problems are harder on the input side.)

Besides processing images and drawings in more-or-less straightforward ways, networks could be used for much more sophisticated manipulations. For example, a network could be trained to take an image of a face as input, and produce as output an image of that face now displaying any of

several chosen emotions: smiling, frowning, staring, tongue stuck out, etc. Cottrell and Metcalfe (1991) and Fleming and Cottrell (1990) have investigated networks for face, gender, and emotion recognition, and have found that to achieve these tasks, the networks develop internal representations of the input faces which could be manipulated at the output level to alter those faces in systematic ways. This opens up the possibilities for a wide range of very powerful image manipulation techniques, altering scenes in meaningful ways, putting people and animals in different poses, changing the bloom on plants, etc.

From there, it is only a small step to the manipulation of \fImoving\fP images, especially represented as successions of static frames. Neural networks could straightforwardly be trained to perform sophisticated tweening between frames in animation, for instance. Sequential networks could be used to produce the step-by-step movements of objects following some trajectory in space, while the types of networks just described could adjust the expressions and gait of animated characters (see de Garis, 1991, for a network that controls the walking of an animated creature).

But even more exciting would be to let neural networks control the actions and behaviors of actors in a visually-displayed world entirely on their own, without following a preordained animation script. By imbuing artificial creatures in a virtual world with the ability to move, sense, eat, fight, hunt, court, etc. independently, we could create "living paintings" in which we watched a herd of sheep grazing and moving about on a landscape, or the interactions of two birds in flight, or the slow and steady movement of an ant colony across a jungle floor. Researchers in the field of artificial life (Langton et al., 1992) are beginning to realize some of these goals. The "brain-style computation" of neural networks is a very natural choice for simulating a creature's behavioral and cognitive mechanisms (Miller and Todd, 1990). Ackley and Littman (1992) have developed a system in which neural network-controlled creatures roam around a two-dimensional world, looking for food, avoiding obstacles, dodging predators who also hunt them down, and otherwise acting the way real organisms do; when these interactions are displayed on a computer screen in real-time, they are very compelling and engaging. Networks could also be used to control independent and realistically-behaving simulated "pets" and other creatures that a human could actually interact with in a virtual reality system. Here again, obviously, the creative angles are endless.

Networks can also find interesting use in real-world kinetic applications, for controlling the motion of robots and vehicles from single arms to self-driving automobiles (Miller, Sutton, and Werbos, 1990). Pomerlau (1991) uses networks to guide the navigation of an autonomous land vehicle (ALVINN) by visual tracking of a road surface. Mel (1990) has developed a system (MURPHY) that controls robot arm motion based on visual input of the arm and goal's location. Pole-balancing, in which a movable platform (a cart) is shifted around to keep a pole balanced on it remaining upright, is another common control problem to which networks have been successfully applied. Artforms including kinetic sculpture, machine dance, automated music conducting, etc., are fertile areas for neural network applications.

Finally, as mentioned earlier, neural networks can be used in human-computer interaction, to help create interactive artworks by following the performance and instructions of human artists, dancers, conductors, etc. in real time. Preliminary work has been proposed for network processing of gestural inputs from Powerglove-like input devices, and from Max Mathews' radio drums; virtually

any dynamic input of this sort (even EEG signals in one application!) is amenable to processing by a neural network which produces appropriate interpreted commands as its output. With speech and handwriting recognition capabilities as well, networks may be able to provide unprecedented levels of communication back and forth between artist and computer.

Conclusions

The current status of neural networks in the arts is one of great promise, but little realization. At best, networks can presently be used as intelligent composer's aids in the creation of new pieces of music, based on learned examples. This is a valuable service, and one at which the network, as an artist's apprentice, can get better and better as it learns more from the artist. Such a support role is probably how neural networks will first appear in applications in other artistic domains as well, generating small ideas and variations which the human artist can then incorporate or disregard in the most aesthetic fashion.

There is some debate over whether the neural network approach to art, and in particular music, is an appropriate one (see Todd and Loy, 1991, section IV). Some argue that the reason computers have been so valuable in art is precisely that they've allowed us to imagine processes by which we'd like to create artworks, specify a very precise set of rules to carry out that process, and then implement those rules to fulfill our imagination. Neural networks, they argue, have eliminated that ability, and chained us back down to only those artforms and styles that we can collect examples of. But this argument, of course, is invalid--if we can create an algorithm to generate new artistic styles in the traditional manner, then we can use those computer-generated works as further examples on which to train our networks. Thus networks need not constrain us at all, but rather add yet another tool into the artist's toolbox. And it is ultimately the products of those tools that we will judge artistically, not the tools themselves.

(For more information on a variety of types of neural networks and their applications, see the yearly proceedings of the Neural Information Processing Systems (NIPS) and International Joint Conference on Neural Networks (IJCNN) meetings, from which several of the papers in the references are taken.)

References

- Ackley, D., and M. Littman (1992). Interactions between learning and evolution. In Langton, et al.
- Bharucha, J.J. (1991). Pitch, harmony, and neural nets: A psychological perspective. In Todd and Loy.
- Chen, J.R., R.K. Belew, and G.B. Salomon (1990). A connectionist network for color selection. In M. Caudill (Ed.), *Proceedings of the (January) 1990 International Joint Conference on Neural Networks*, vol. II. Hillsdale, NJ: Lawrence Erlbaum Associates, pp. 467-470.

Cottrell, G.W., and J. Metcalfe (1991). EMPATH: Face, emotion, and gender recognition using holons. In R.P. Lippmann, J.E. Moody, and D.S. Touretzky (Eds.), \fIAdvances in neural information processing systems 3.\fP San Mateo, CA: Morgan Kaufmann Publishers, pp. 564-571.

de Garis, H. (1991). Genetic programming. In B. Soucek (Ed.), \fINeural and intelligent systems integration.\fP Wiley.

Desain, P., and H. Honing (1991). The quantization of musical time: A connectionist approach. In Todd and Loy.

Dolson, M. (1991). Machine tongues XII: Neural networks. In Todd and Loy.

Fleming, M.K., and G.W. Cottrell (1990). Categorization of faces using unsupervised feature extraction. In \fIProceedings of the (June) 1990 International Joint Conference on Neural Networks,\fP vol. II. Ann Arbor, MI: Edwards Brothers/IEEE, pp. 65-70.

Gjerdingen, R.O. (1991). Using connectionist models to explore complex musical patterns. In Todd and Loy.

Grebert, I., D.G. Stork, R. Keesing, and S. Mims (1991). Connectionist generalization for production: An example from GridFont. In \fIProceedings of the (July) 1991 International Joint Conference on Neural Networks,\fP vol. II. Piscataway, NJ: IEEE Inc., p. 105-109.

Jordan, M.I. (1986). Attractor dynamics and parallelism in a connectionist sequential machine. In \fIProceedings of the Eighth Annual Conference of the Cognitive Science Society.\fP Hillsdale, NJ: Erlbaum Associates.

Kohonen, T., P. Laine, K. Tiits, and K. Torkkola (1991). A nonheuristic automatic composing method. In Todd and Loy.

Langton, C.G., C. Taylor, J.D. Farmer, and S. Rasmussen (Eds.) (1992). \fIArtificial Life II.\fP Redwood City, CA: Addison-Wesley.

Lewis, J.P. (1991a). Creation by refinement and the problem of algorithmic music composition. In Todd and Loy.

Lewis, J.P. (1991b). Probing the critic: Approaches to connectionist pattern synthesis. In \fIProceedings of the (July) 1991 International Joint Conference on Neural Networks,\fP vol. I. Piscataway, NJ: IEEE Inc., p. 85-90.

Lippmann, R.P. (1989). Review of neural networks for speech recognition. \fINeural Computation, 1,\fP 1-38.

McClelland, J.L., and D.E. Rumelhart (1988). \fIExplorations in parallel distributed processing.\fP Cambridge, MA: MIT Press/Bradford Books.

Mel, B.W. (1990). Vision-based robot motion planning. In Miller, Sutton, and Werbos.

Miller III, W.T., R.S. Sutton, and P.J. Werbos (Eds.) (1990). \fINeural networks for control.\fP Cambridge, MA: MIT Press/Bradford Books.

Miller, G. F., and P.M. Todd (1990). Exploring Adaptive Agency I: Theory and Methods for Simulating the Evolution of Learning. In D.S. Touretzky, J.L. Elman, T.J. Sejnowski, and G.E. Hinton (Eds.), \fIProceedings of the 1990 Connectionist Models Summer School.\fP San Mateo, CA: Morgan Kaufmann, pp. 65-80.

Mozer, M.C. (1991). Connectionist music composition based on melodic, stylistic, and psychophysical constraints. In Todd and Loy.

Pomerlau, D.A. (1991). Rapidly adapting artificial neural networks for autonomous navigation. In R.P. Lippmann, J.E. Moody, and D.S. Touretzky (Eds.), \fIAdvances in neural information processing systems 3.\fP San Mateo, CA: Morgan Kaufmann Publishers, pp. 429-435.

Rumelhart, D.E. (1990). Brain style computation: Learning and generalization. In S.F. Zornetzer, J.L. Davis, and C. Lau (Eds.), \fIAn introduction to neural and electronic networks.\fP San Diego: Academic Press.

Rumelhart, D.E., G.E. Hinton, and R.J. Williams (1986). Learning internal representations by error propagation. In Rumelhart and McClelland.

Rumelhart, D.E., and J.L. McClelland (Eds.) (1986). \fIParallel distributed processing: Explorations in the microstructure of cognition. Vol. 1: Foundations.\fP Cambridge, MA: MIT Press/Bradford Books.

Sano, H., and B.K. Jenkins (1991). A neural net model for pitch perception. In Todd and Loy.

Sayegh, S.I. (1991). Fingering for string instruments with the optimum path paradigm. In Todd and Loy.

Todd, P.M. (1991a). A connectionist approach to algorithmic composition. In Todd and Loy.

Todd, P.M. (1991b). Further research and directions. In Todd and Loy.

Todd, P.M., and D.G. Loy (Eds.) (1991). \fIMusic and Connectionism.\fP Cambridge, MA: MIT Press.

Trubitt, D.R., and P.M. Todd (1991). The Computer Musician: Neural Networks and Computer Music. \fIElectronic Musician, 7(1),\fP 20-24.