NETZSPRECH - Another Case for Distributed 'Rule' Systems

Georg Dorffner
Dept.of Medical Cybernetics and Artificial Intelligence
University of Vienna, Austria
and: Austrian Research Institute for Artificial Intelligence

Abstract: This paper compares conventional symbolic rule systems with distributed network models, considerably arguing for the latter. NETZSPRECH - a network that transcribes German texts similar to NetTalk is first introduced for this purpose and serves as an example for the arguments.

1. Introduction

Models in artificial intelligence (ai) and cognitive science rely mostly on the assumption that cognitive systems (such as the human brain) cannot only be described but also efficiently modeled using symbolic descriptions and rules to combine them. However, many aspects of human perception and cognition are left out by those systems, which can be attributed to a great extend to the limitations of symbol-and-rule systems due to their brittleness. Recently, research on parallel distributed processing (PDP) models has shown that those models can have the power to overcome some of the said limitations. Nevertheless, there still exits many critics and sceptics toward the PDP paradigm (such as in Pinker&Prince 1987) who neglect the points where PDP models can be superior.

The purpose of this paper is to describe Netzsprech, a new implementation of the Nettalk model by Sejnowski&Rosenberg (1986) adapting it to the German language, and to use this model to argue for PDP models and against models merely based on symbols and rules. Netzsprech is a network that (similarly to Nettalk) learns to translate text into a phonetic transcription from examples, i.e. it learns to pronounce German words. Reading and pronouncing a script is a process that appears suitable to show the advantages of a distributed model over a symbolic one.

2. The Netzsprech model

Netzsprech is a basic so-called associative network (AN) (Rumelhart, McClelland 1986) consisting of three layers of units. Many of the processes that can be modeled by an AN appear to be rule-governed in that symbolic rules can describe the association to a large extent. Thus, ANs can be viewed as distributed 'rule' application systems. I put 'rule' under quotes here, because it has to be understood in a much more relaxed sense than rules in conventional AI systems. Distributed 'rule' application can overcome many of the restrictions symbolic rule application is bound to. The architecture of Netzsprech is depicted in Fig.1.

A perfect example of seemingly rule-governed behaviour with

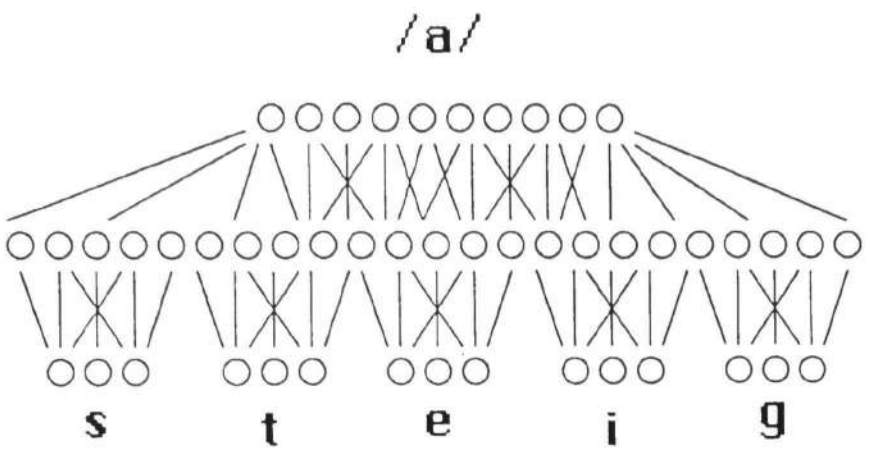

Fig.1

nevertheless a lot of exceptions is the human ability to read text aloud, that is, to transform the written representation of a text to the phonemic pattern that is to be spoken. T.Sejnowski and R.Rosenberg (1986) have shown in an impressive manner that their NetTalk model could learn to transcribe English text into phonemes, which were then fed into a phonemic speech synthesizer. Without doubt, English is the Indoeuropean language with the most discrepancies between graphemic and phonemic patterns. Nevertheless, NetTalk could learn to pronounce simple English text with an error rate well below 10%.

This paper describes a network that learns the pronounciation of German words in a very similar fashion to NetTalk. Hence, I called it 'Netzsprech'. With this implementation I want to demonstrate the advantages of distributed learning and representation schemes which led me to believe that those schemes will be a necessary complement for AI programs in the future.

3. What Netzsprech does

The input layer (fig.1) is divided into five clusters which encode five letters of German text. The letter in the middle is the one to be transcribed, the two on the left and on the right, respectively, serve as context information to aid the transcription. Sejnowski and Rosenberg (1986) used seven letters (i.e. a context of three letters on each side), but as it turns out, in German generally a two-letter context is sufficient, if one excludes difficult foreign words.

The representation of letters in the five input clusters is a local one, that is, one unit in each cluster, when activated, corresponds to exactly one letter in the text. As a result of this, only one unit can be active per cluster at any given time. As there are 31 letters (including umlauts, the 'scharfe s', and a space), the input layer consists of 5 times 31 units.

For representing (encoding) the phonemes in the output layer a

different strategy was used. Here one expects to find similar codes for similar phonemes. One example in German is the letter 'd' that can be pronounced as a [d] (in a Wort like 'der') or as a [t] (at the end of a word like 'und', this is known as final devoicing in German). Although there are two different phonemes for the letter 'd', these phonemes are nonetheless very similar to each other (in fact, they only differ in their voicedness). For the model to be adequate, these two similar phonemes should be encoded in two similar output activation patterns. This was done using a binary code distributed over the activations of 10 output units.

The hidden layer which serves to provide the possibility to learn arbitrary pattern mappings between input and output, consists of 30 units.

4. The training phase

The training set consisted of a list of the 1000 most common words in German (according to Meier (1978)) plus their transcription. The transcription that was available in machine-readable form (produced by a rule-based model designed by Pounder and Kommenda (1986)) had to be hand-edited to take account of the fact that the Netzsprech network requires a one-to-one mapping between letter and phoneme. Thus, when there were cases where a whole sequence of letters has to be pronounced as a single phoneme, the phoneme was taken as the transcription of the first letter. The other letters were transcribed as 'silent' (using a pseudo-phoneme [+]). For example, the sequence /sch/ has to be pronounced as [ ] in most cases, so that the transcription of the word 'asche' became [a ++$] (where [$] is the schwa-phoneme). The reverse case - one letter has to be pronounced as a sequence of phonemes - does not occur in standard German, if one writes affricates (like [ts] for the letter /z/) as one phoneme.

Training consisted of taking each word from the list (in the order they appear) and its transcription and presenting it to the network, letter by letter, encoding both letters and phonemes. After each presentation (5 letters plus a phoneme) the delta rule was applied and the weights were adjusted.

5. The results

Fig.2 shows the learning curve of the network. The x-axis shows the number of words used for training and the y-axis the number of correctly transcribed letters and words. Presentation of 3000 to 4000 words (that means, the list of 1000 words was scanned 3 to 4 times) is sufficient to bring the network very close to convergence, with the error rate for letters going down to less than 10 %. Most of the errors that remain at this point are minor, i.e. only 1 to 2 of the 10 featurse are produced incorrectly. The understandability (the phonemes were subsequently synthesized) is already extremely high.

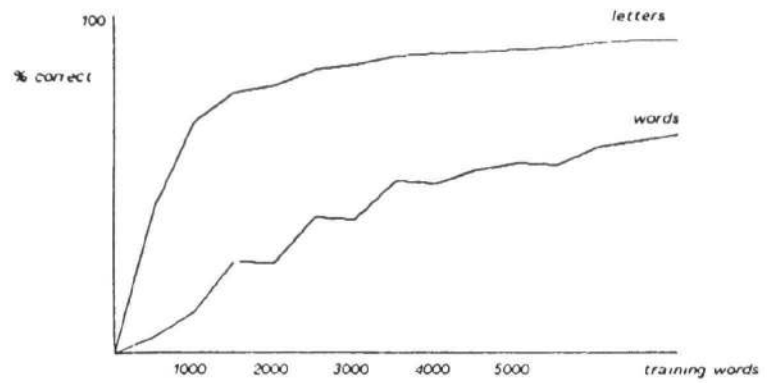

Fig.2

6. Discussion of the network behaviour

At no point in this paper do I want to claim that the model I am describing is a valid model for the complex process of learning how to talk (neither did Sejnowski&Rosenberg). Nevertheless, the model shows some very important behaviour that - if conceived as part of a much more complex model - seems to reproduce many aspects of human processes of understanding and learning in a very plausible manner.

The problem Netzsprech was faced with is one that, to a large extent, is describable with a symbolic rule system, containing rules like

(1) #s{p,t} -> [ ] (2) s -> [s]

meaning that 's' is to be pronounced as an [s] by default and as an [ ] at the beginning of a word (marked by '#') in front of a 'p' or 't'. However, as I am going to show, the distributed system in Netzsprech is superior to the rule model in that it corresponds more closely to human processes and has the advantage of being easily learnable and adaptable.

6.1 Rules are applied simultaneously

One of the weak points of symbolic rule systems is that they almost always involve search or other serial processes. For example, if there exist pronounciation rules for every letter in German, then in a first step all rules for the letter currently considered have to be found. In a second step, all rules among them have to be chosen that match the current context ('matching cycle'). Of course, rule matching could be done in parallel, which reduces this search process to one step. But after that, the system most of the time is faced with the problem of conflict resolution, the selection of the rule to apply among several that all match the context. This can be the case with the two rules given above, in a context like '_ # s p i'. The time it takes to solve problems of rule conflict increases when we add more rules, which is counter the experience that humans

use less time with additional knowledge rather than more.

To get around the problem of rule conflict, the rules could be written in such a way that only one rule applies per context. In other words, a rule would have to be written for virtually every possible context. For example, there would have to be a rule for 's' in front of 'p', where it is pronounced as an [ ], but also one for 's' in front of every other consonant, where it is pronounced as an [s]. For the domain being discussed this might not be impossible to do, but problems are conceivable, where the possible space of contexts is so large that it becomes very inefficient if not impossible to provide a rule for every context.

But even if it were possible in every case, one important property of the original rule scheme is lost: One can no longer say, what is the 'default' rule for a given letter, for example, that 's' in the majority of the cases is pronounced as [s] and therefore this output is the most likely one. In the case where we had two (sometimes competing) rules, by virtue of the lowest priority one of the rules was the default. If little or no context was specified, this rule applied and gave the most likely output.

Thus, in rule based systems, there is a trade-off: Either the system can easily generalize and provide default outputs for unknown inputs, but then it shows rather implausible processing time relations. Or the system can produce the output more plausibly, but then the power for generalization and default outputs is lost.

A distributed system like Netzsprech, however, exhibits both advantages at the same time. All the distributed 'rules' are applied in parallel, so there is never anything like conflict resolution, let alone serial matching processes. Also, as the size of the network is fixed, acquiring new knowledge cannot make the process slower. At the same time, the model shows full power of generalisation for inputs it has never seen before with an inherent default mechanism. Consider the following example from Netzsprech: If an 's' is presented to the trained network without any context (i.e. all context units are set to 0, note that this case never occured during learning) the default (or most likely) output, [s], is activated. Where no such default exists, for example for an 'x' (a very rare letter in German proper), the output is unspecified (i.e. some arbitrary output which nevertheless is closest to the most frequent phoneme corresponding to 'x'). If now, in the case of the 's', a context is added that in German should alter the pronounciation (like a '#' in front and a 'p' afterwards), the output in fact changes to (the now default case) [ ]. One can see from these examples, that the model incorporates both default and more specififc rules at the same time. The context changes something in the output only when it, together with the central letter,

constitutes a case for a more specific rule to the default.

6.2 The output has continuous values

A major property of a distributed network is that it allows for activation values along a continuous scale, indicating how strong a specified hypothesis is. Returning to the above example, when 's' is presented alone to Netzsprech, the output is the default, an [s]. However, the strength of the output is lower than it is when 's' is presented in the more specific context 'ase'. The reason is that in the former case, without any further specification, the 's' is pronounced as an [s] only with a certain probability, if the (unknown) context were '#sp', then it could also be a [ ].

This property of the system can be useful when incorporating the network into a bigger one. For example, one could conceive of another component that controls the movements of the articulatory organs. This component would receive input from several components; mainly, of course, from Netzsprech, but other inputs that influence the movement of the articulators would be possible, too. If now the input from Netzsprech has a rather low level, other sub-systems could more easily 'push' the control component into another direction. For example, even when Netzsprech says [s] the actual pronounciation might turn out to be [ ], when the output was a default case. In the more specific case, with higher output activations, this scenario is less likely. Thus, continuous output values are important for modeling contextual and con-situational effects.

In a symbolic rule system, it is much more difficult to obtain such behaviour. Symbolic rules are much more isolated from each other and from the rest of the system. Making them influence each other is much harder and might not even be efficiently possible. The advantage of the network model is that it provides easy entry points - the units and connections themselves - for other system components to modify the behaviour.

6.3 Rules are learned according to environment

Although the behaviour of the network after learning appears to be rule-governed (such as: a feature F in pattern A produces a feature G in pattern B), none of the 'rules' has to be stated explicitely or beforehand. The network can 'find out' the rules (or whatever we want to call them) by itself from the examples, which then cover exactly the cases in the training set and those derived through extrapolation of the rules. In other words: No one has to stipulate the r presented. Furthermore, as Rumelhart (1986) points out, each 'rule' currently incorporated in a PDP model is applied with exactly the strength that corresponds to its fitness.

6.4 Networks are robust

In a symbolic rule system it is crucial that at any point in time all the needed rules exist and are accesible. When one of them is missing, the system might fail completely. In a network this can never happen. As the 'rules' are represented in a highly distributed manner and - in addition - are all learned from the environment, they cannot easily get lost, not even by partial destruction of the system.

7. Conclusion

Much more could be said about distributed 'rule' systems (Dorffner 1987, 1988). However, this short discussion should already have made clear the advantages of such models. The results suggest that to model cognitive processes such as understanding and producing natural languages - which are much more complex than the task of Netzsprech but largely based on similar mechanisms of association - one will have to include PDP or related models in the future.

References

Dorffner G.: NETZSPRECH: A Network Learns German - Description of the Model and Discussion, Tech.Report 87-01, Dept.for Med.Cybernetics and AI, University of Vienna; 1987.

Dorffner G.: Modeling Gestalt Phenomena in a Distributed 'Rule' System, in: Trappl (ed), Cybernetics&Systems '88, Kluwer Academic Publishers; 1988.

Hinton G.E., McClelland J.L., Rumelhart D.E.: Distributed Representations, in Rumelhart & McClelland (1986).

Pinker S., Prince A.: On Language and Connectionism: Analysis of a Parallel Distributed Processing Model of Language Acquistion, Dept.of Brain and Cognitive Science, MIT, occ.paper #33; 1987.

Pounder A., Kommenda M.: Morphological Analysis for a German Text-to-Speech System, in Proceedings of the 11th International Conference on Computational Linguistics (COLING-86), Bonn, FRG; 1986.

Meier H: Deutsche Sprachstatistik, Georg Olms Verlag, Hildesheim, 1978.

Rumelhart D.E., McClelland J.L.: Parallel Distributed Processing, Explorations in the Microstructure of Cognition, Vol 1&2. MIT Press, Cambridge, MA; 1986.

Sejnowski T.J., Rosenberg C.R.: NETtalk: A Parallel Network that Learns to Read Aloud, Johns Hopkins University, Tech.Rep. JHU/EECS-86/01; 1986.