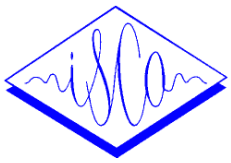

The ESCA Workshop
on Speech Synthesis
Autrans, France
September 25-28, 1990

TEXT-TO-PHONETICS TRANSLATION WITH SYNTACTIC NEURAL NETS

S.M. Lucas and R.I. Damper
Department of Electronics and Computer Science,
University of Southampton, Southampton SO9 5NH, UK

Abstract

This paper shows how syntactic neural networks can be applied to the problem of translating orthographic strings to phonetics strings. The work has two novel aspects. First, the model is symmetric and so is also capable of phonetics-text translation. Second, although training is based on a set of whole-word orthographic/phonetic symbol-string pairs, it is unsupervised in the sense that no segmentation information is included. The training data consists of a (randomly-selected) subset of N mono-syllabic pairs extracted from the machine-readable Oxford Advanced Learners' Dictionary. The trained nets were tested on the training subset and an equal size (disjoint) test-set. Early results show that translation accuracy – as assessed by the Levenstein distance between the network's output and dictionary transcription – is asymptotic to 50%, for *both* seen and unseen words, as N increases.

1 Introduction

Early attempts at text-phonetics translation (e.g. Ainsworth, 1973) aimed at defining an appropriate translation-rule formalism, before setting about the task of rule-writing with the aid of an expert. Although such systems have achieved useful results, their creation is a tedious process and the rule-set is difficult to modify in the light of errors. Recently, interesting advances have been made in automating the process. Probably the most famous example of this is NETtalk (Sejnowski and Rosenberg, 1987). This involves training a multi-layer perceptron in a supervised manner to associate a window (length 7) of orthographic characters with the correct phonemic output vectors. The learning algorithm used is error back-propagation.

We describe a new approach to the problem using cross-coupled syntactic neural networks (SNNs). Training is as unsupervised as possible; the nets are simply presented with *whole-word* text-phonetics pairs. They must learn not only translation probabilities, but also the actual *units* of translation. Thus, while the net may decide that entire syllables are appropriate in some cases, on other occasions single letters will be used as the basis for translation. SNNs operate with an underlying grammar, in this case stochastic context-free. The cross-coupled SNNs are formally equivalent to a syntax-directed translation scheme (Aho, Sethi and Ullman, 1986) but with the stochastic nature of the grammars providing additional power. In the same way, the use of stochastic productions has advantages over conventional, rule-based translation which use techniques such as rule-ordering to overcome their inability to handle probabilities.

2 Statistical Models of String Translation

Our starting point is a lexicon of text-phonetic word-pairs. If this covered all possible words that we might ever encounter and we had enough memory, we could do no better than to use this as a look-up table to perform translation. In practice, however, we will encounter novel words. The assumption we make is that these novel words should be pronounced in a way that is statistically similar to the words in the training lexicon. The question now reduces to: what statistics should we measure, and how?

The solution we adopt here is most easily visualised as a three-pass procedure. First, we enumerate the most probable substrings in each domain starting from the alphabets of atomic symbols, Σ_o and Σ_p where o and p refer to orthography and phonetics respectively. The most probable concatenations of these are added to the set of non-terminals (initially empty) to form new alphabets of symbols, N_o and N_p. In grammatical terms, the original (atomic) symbols correspond to the terminals, while concatenations of these correspond to non-terminals.

At this stage, we have an alphabet $A_o = \Sigma_o \cup N_o$ of orthographic symbols and, likewise, A_p for the phonetic symbols. The second pass simply calculates the bigram statistics of these symbols in their respective domains. Note that this is much more powerful than *ordinary* bigram statistics, since some of the symbols correspond to several atomic characters.

The final pass calculates the translation probability $P(o \rightarrow p|o)$ i.e. given $o \in A_o$, the probability that it rewrites to this particular $p \in A_p$. Currently, all possible translations (given A_o and A_p) are produced, typically between 3 and 20 for each word-pair, and the network trained on these.

3 Syntactic Neural Networks for String Translation

Space only allows a brief account of SNNs here; for details see Lucas and Damper (1990a,b). The important points are that SNNs have interpretations in formal grammars and that learning may be unsupervised. Training is a process of grammatical inference while recognition is parsing with respect to the inferred grammar. The underlying grammar allows the net to be used generatively, making it possible to use two coupled SNNs as a complete translation system. The basic set-up is shown in Figure 1. Importantly, our model is dynamic – input flows into the orthographic net (say) and, as time passes, it begins to flow out of the phonetic net (running in generation mode). Thus, it is able to translate strings of any length.

During training, the orthographic-phonetic word-pairs are presented to the network, one word at a time. Consider first a single net. As each input string passes through the SNN's input window (of length 1), the lowest level of the net observes both the current symbol, and the previous one, which has gone through a unit time delay. If this particular symbol pair occurs often enough, a new neuron will be allocated to represent it; in grammatical terms, this corresponds to the creation of a new non-terminal, and a new rule which rewrites that non-terminal as the two lower-order symbols. This rule-gathering process extends up as many layers as necessary, the criterion being the probability of the event compared to a predefined threshold. In most SNN applications, we devise clustering methods to amalgamate these rules, thereby reducing the number of non-terminals necessary. We have not done this here, partly because of the difficulty of developing sensible distance metrics in such distinctly symbolic domains but also because it does not seem necessary; the network performs at a useful level without becoming too large.

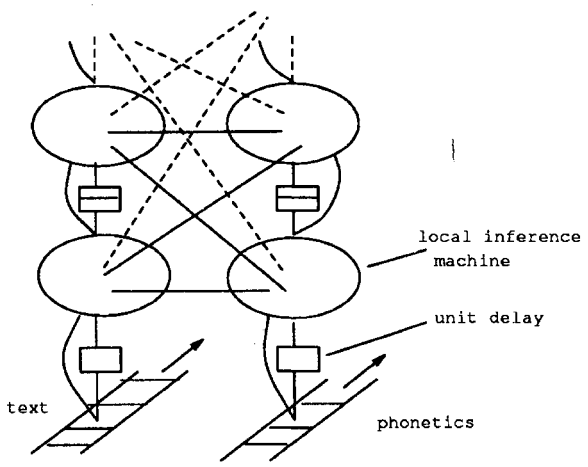

Figure 1: Two cross-coupled syntactic neural networks. One network is dedicated to looking for regularities in the orthography, while the other analyses the phonology. During training, the cross-coupling weights are adjusted to associate related events in the respective domains. During translation strings are fed in to the required network. This causes excitations in the coupled network, which is used generatively to produce output symbols in its input domain. Hence a translation is produced.

In SNNs, the connection weights corresponding to each rewrite rule are normally adjusted to the probability of that rule firing, given that we have the non-terminal. Since the rules are not agglomerated, each of these probabilities is now 1. Here, however, there are the cross-coupling and recurrent weights to be considered. These are used to compute the translation probabilities (of a phonetic symbol given an orthographic symbol, or vice versa) and the within-domain transition probabilities of the next symbol given the current one, respectively. By symbol, we mean *terminal* or *non-terminal*. This is important, since a non-terminal may span many terminals. Thus, in addition to the translation probabilities, we have an adaptive n-gram model, where we may predict the next symbol according a varying number of previous symbols, depending on their joint probability. Intuitively, this makes good sense; if the probability of an n-gram is low, then any predictions of the next symbol based on it will be unreliable – we might just as well only consider the more probable ones.

4 Results

Figure 2 shows translation accuracies obtained thus far, using randomly-selected samples of mono-syllabic word-pairs drawn from the machine-readable dictionary. Each pair was of the form ((zoom)(zu:m)), where the first element is the orthography and the second is the phonology. $|A_o|$ and $|A_p|$ were limited to 200.

Results are based on two measures. The first (*word*) considers the translation incorrect it matches the dictionary entry exactly. The second (*symbol*) is the unweighted Levenstein distance i.e. the number of string-edits (insertions, deletions, substitutions) required to convert the output to the correct translation, normalised by the maximum possible number of such edits given those string lengths. By "unweighted", we mean that all different symbols are assumed equally dissimilar. Typically, translation time was about 1 word/s while the training rate was about 20 word/s. The software is in prototype stage, howevere, and the translation process in particular could be made considerably faster with a little effort.

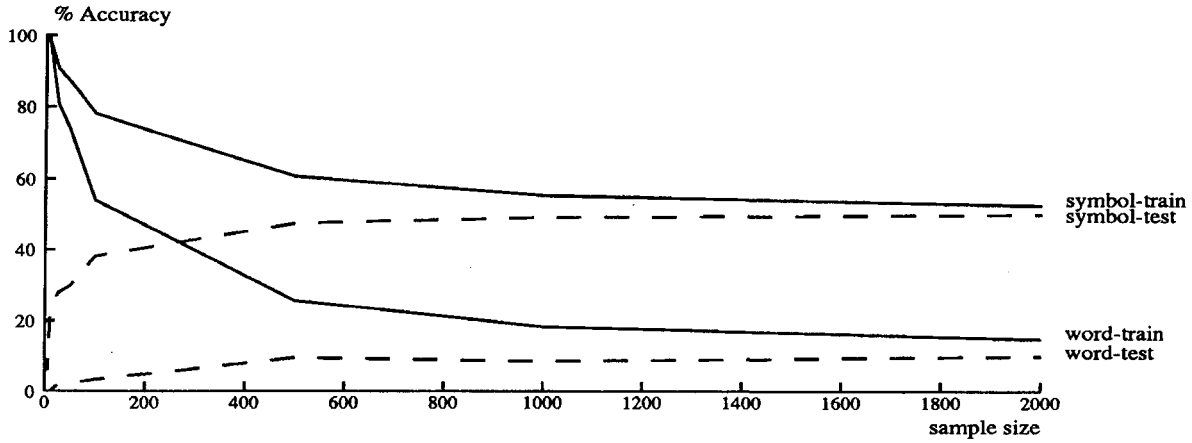

Figure 2: Percentage accuracy plotted against sample size. In all cases, the test-set size was equal to the training-set size. See text for explanation of *symbol* and *word* scores.

There are several straightforward ways in which results could be improved, such as adding spaces to distinguish word start and end. Also, we could increase $|A_o|$ and $|A_p|$ beyond the present, arbitrary limit of 200. The training and test word-pairs could be frequency-weighted according to some corpus. The current case where all words are assumed equiprobable is the most difficult, having the largest possible entropy (at least, to a first-order approximation).

5 Conclusion

We have introduced a statistical model of string translation and outlined how it can be implemented in connectionist fashion, within the paradigm of syntactic neural networks. The model is unusual in two respects; its symmetry and the unsupervised nature of its training. Present results are poor compared to conventional translation systems, but most of the errors may be due to the system's immaturity. We would expect the changes outlined above to improve performance considerably. Given the claimed symmetry of the model, we intend in future work to evaluate its potential for translating phonetics to orthography.

References

AHO, A.V., SETHI S. AND ULLMAN, J.D. (1986) *Compilers: Principles, Techniques and Tools.* Addison Wesley, Wokingham, England.

AINSWORTH, W.A. (1973) "A system for converting English text into speech", *IEEE Transactions on Audio and Electroacoustics*, **AU-21**, 288 – 290.

LUCAS, S.M. AND DAMPER, R.I. (1990a) "Syntactic neural networks", *Connection Science*, in press.

LUCAS, S.M. AND DAMPER, R.I. (1990b) "Self-organising temporal networks", *Proceedings of IEEE Workshop on Genetic Algorithms, Neural Networks and Simulated Annealing Applied to Problems in Signal and Image Processing*, Glasgow, Scotland.

SEJNOWSKI, T.J. AND ROSENBERG, C.R. (1987) "NETtalk: a parallel network that learns to pronounce English text", *Complex Systems*, **1**, 145 – 168.