AUTOMATED GENERATION OF N-BEST PRONUNCIATIONS OF PROPER NOUNS

Neeraj Deshmukh, Mary Weber, Joseph Picone

Institute for Signal and Information Processing
Department of Electrical and Computer Engineering
Mississippi State University, Mississippi State, Mississippi 39762
{deshmukh, weber, picone}@isip.msstate.edu

ABSTRACT

The problem of proper noun recognition is key to developing pervasive voice interfaces in applications such as directory assistance and data entry for telecommunications. Recognition of such words requires an ability to generate reasonably accurate pronunciation networks. This is a very challenging problem because a large percentage of proper nouns, such as personal names, appear to have no obvious (or simple) letter to sound mapping rules that can be used to generate the pronunciations. It appears to be an open-ended problem that is constantly evolving as a function of numerous sociological factors. Yet humans do amazingly well at generating and recognizing the pronunciation of a name never encountered before. We present an algorithm based on a Boltzmann machine type of neural network that generates the most likely pronunciations of a proper noun from the text-only spellings of the name. This method does not require voice data containing the spelling or nominal pronunciation.

1. INTRODUCTION

As the voice interface market has grown, so has the demand for simple, intuitive, and user-friendly interfaces. In many applications, particularly those related to medicine [1, 2], the ability to recognize a physician's or patient's name is crucial in providing a usable interface. A comparable problem involving company names and product names exists in voice interfaces for advanced telecommunications services.

Traditional systems for proper-noun pronunciation employ extensive hand-written rule sets to generate an accurate pronunciation. Such systems, even though deemed accurate for the directory assistance application for which they were designed, essentially solved an ill-posed problem. Moreover, they claimed to generate only the single-most likely pronunciation of a name. This obviously is not the most useful approach for a speech recognition application. Since most proper names have a number of highly probable pronunciations that can rarely be differentiated from the context of the application, it is important that all plausible alternatives be available to the recognizer.

An alternative approach is to use massively-parallel network models [3,4]. Knowledge in such connectionist systems is distributed over multiple processing units and the net exchange of information between these units determines the behavior of the network. Multi-layered neural networks, in which the internal or hidden units can act as feature detectors that perform a mapping between the input and the output are a class of models ideally suited for such applications.

The system we present in this paper relies on a particular form of neural network designed to generate multiple outputs for a given input — a Boltzmann machine [5]. This network transforms its input, the spelling of a proper name, to a network of distinctive features describing articulatory movements required to produce various pronunciations of the name. Recently, similar work involving Hidden Markov Model-based approaches has also shown some promise [6].

2. BOLTZMANN MACHINE

The Boltzmann architecture is designed to allow efficient searches for combinations of hypotheses that maximally satisfy the input data and some stored constraints. Each hypothesis is represented in terms of a unit in the network whose binary state represents the truth values of the hypothesis. The interaction between such units implements the stored information about the constraints. Data is supplied to these units in the form of external inputs. Constraints are modified by altering the interaction weights.

The Boltzmann machine is a parallel computational architecture [Figure 1] quite similar to a Hopfield network [7]. We can assign each global state of the network a numerical *energy* value, and then make the individual units act to minimize the global energy compatible with each input configuration. The energy of the network is defined as

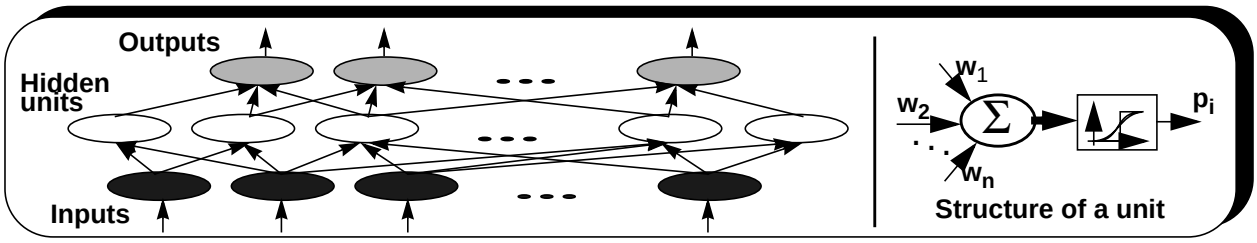

Figure 1. A Boltzmann Machine.

$$E = -\sum_{i<j} w_{ij} s_i s_j + \sum_i \theta_i s_i \quad (1)$$

where w_{ij} is the weight of the link between the units i and j, s_i is 1 if the unit i is *on* and 0 otherwise, and θ_i is a threshold, as shown in Figure 1. This energy value can be interpreted as a metric of the system deviation from the constraints implicit in the data (or alternately, a measure of the mismatch between the statistics of the input data and the statistical model represented in the machine). By minimizing the energy the system is forced to evolve in a fashion that progressively satisfies these constraints.

2.1. Minimization of Energy

A simple technique to minimize the global energy function is to switch each individual unit to a state that results in a lower energy value given the current states of the other units [7]. The global *energy gap* of the system at each unit is the difference in energy with the unit hypothesis accepted and rejected, and is given for the i^{th} unit by

$$\Delta E_i = \sum_j w_{ij} s_j - \theta_i \quad (2)$$

Thus by adopting the *on* state if the total weighted input to a unit exceeds the threshold, we get the familiar decision rule of binary thresholding.

2.2. Finding the Global Energy Minimum

Binary thresholding ensures that the system comes to rest in some local energy minimum. However, this is not an optimal state of the machine for constraint specification. It is possible to escape from poor local minima by allowing the network to occasionally jump to states of higher energy. This is achieved by employing the simulated annealing technique [8] of stochastic relaxation. If the energy gap between the *on* and *off* states is ΔE_i, then regardless of its previous state the probability of a unit turning *on* is

$$p_i = \frac{1}{1 + e^{(-\Delta E_i)/T}} \quad (3)$$

where T is a parameter that acts like temperature. Therefore the relative probability of a system in *thermal equilibrium* to be in the two global states α and β (corresponding to some unit being *on* or *off*) is given by $P_\alpha / P_\beta = \exp((E_\beta - E_\alpha)/T)$ which is the Boltzmann density distribution. An interesting feature of a Boltzmann machine is that the equilibrium distribution uses only locally available information at each unit, even though a local change in the weights optimizes a global measure.

2.3. Training the Boltzmann Machine

The Boltzmann machine is capable of learning the underlying constraints that characterize a problem domain given some examples in that domain, simply by modifying the weights of its interconnections to construct an internal model that is capable of producing similar examples with the same probability distributions. By modifying the weights the machine can be made to approach any desired set of probabilities. This training process is strongly biased towards low energy states at a low temperature, and takes time to achieve equilibrium. At higher temperatures the bias is not so favorable towards energy minima but the learning is faster. A good way trade-off is to begin annealing at a high temperature and gradually cool down; performing a coarse-to-fine search for the global minimum.

2.4. The Learning Algorithm

For an input vector x_i, the output of a hidden layer unit is

$$h_i = \frac{1}{1 + e^{\left(-\sum_j w_{ij}^{xh} x_i\right)/T}} \quad (4)$$

where w_{ij}^{xh} are the weights connecting the input units to the hidden units. The output of the units in the hidden layers is propagated to compute the output of the outer layer units.

$$o_i = \frac{1}{1 + e^{\left(-\sum_j w_{ij}^{ho} h_j\right)/T}} \quad (5)$$

where w_{ij}^{ho} are the weights that connect the hidden layer units to the output layer.The errors in the two layers are

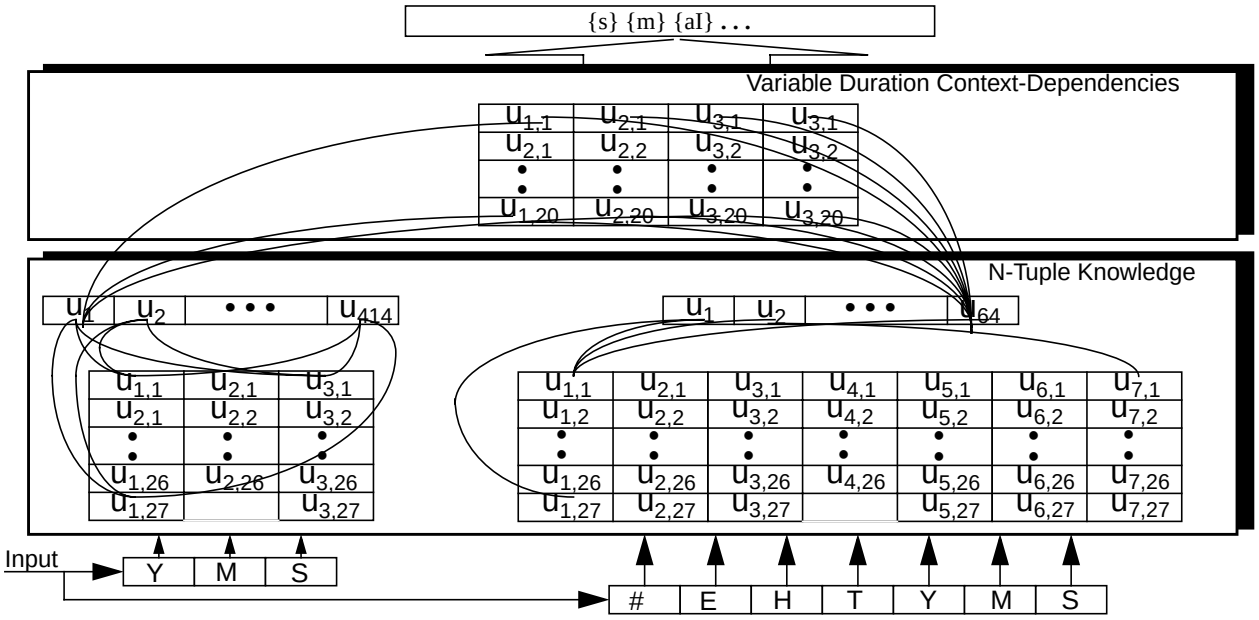

Figure 2. An overview of a neural network architecture that performs letter to sound conversion.

$$\delta_i^{ho} = o_i(1-o_i)(y_i-o_i)$$
$$\delta_i^{xh} = h_i(1-h_i)\sum_j \delta_j^{ho} w_{ij}^{xh} \quad (6)$$

where y_i is the target (or expected) output vector. The weights are updated using these error values with some feedback from the updates in the previous training pass.

$$\Delta w_{ij}^{ho}(t+1) = \eta \delta_j^{ho} h_i + \alpha \Delta w_{ij}^{ho}(t)$$
$$\Delta w_{ij}^{xh}(t+1) = \eta \delta_j^{xh} x_i + \alpha \Delta w_{ij}^{xh}(t) \quad (7)$$

Here η is the *learning rate* and α the *momentum* or feedback coefficient. The machine continues to make passes of the training data till the mean squared error in the output values drops below a suitable threshold. To allow the machine to choose an appropriate direction of learning, the momentum is initialized at a very low value and is gradually increased to approach unity with the epochs.

3. ARCHITECTURE FOR PROPER NOUN PRONUNCIATIONS

The Boltzmann machine architecture used for the pronunciations consists of three principal components: a layer that maps input letters to binary-valued inputs, a layer that maps n-tuples of letters into a set of internal states using the networks context-sensitive knowledge, and a layer that mixes long-term and short-term constraints to refine the interpretations of groups of letters, as shown in Figure 2. This configuration is based on two design criteria:

1. Generally, a relatively small amount of contextual information will be sufficient to narrow the range of possible sound correspondences to a small set.
2. Choosing a correct sound from this set may require information at more remote points in the name.

In a sense, the network is designed to model n-tuples of letters using local and long-distance constraints. The input layer is a shift register structure that is used to buffer characters as they are input to the system one at a time. This approach is similar to other time-delay techniques that have become popular in speech recognition systems. The output layer of the machine consists of units that translate the binary-valued outputs into corresponding phonemes. The connection weights are initialized to small random values and modified during training to model the statistical distribution in the training data.

4. TRAINING DATABASE

As is the case in most forms of speech research, progress is fueled by a comprehensive development database. No appropriate databases currently exist for training the proposed system. Numerous data has been collected anecdotally on the problem of alternate pronunciations, but none of this data has ever been incorporated into a publicly available database.

A significant portion of the effort expended in this work was devoted to the development of a small training

database. We have developed a database of 7,000 surnames that we are using in pilot experiments. We are in the process of expanding the database to include 20,000 surnames and several thousand corporate names. This database adheres to the Worldbet pronunciation standards and represents a reasonably diverse set of names with which we have developed significant experience and a great deal of confidence.

5. EXPERIMENTS AND RESULTS

We performed a number of preliminary experiments with a set of 240 names to study the training phenomena in our network. With closed-loop training on 10 names the machine output accurate pronunciations for all ten. When 100 more names were added for training the machine managed to provide plausible alternative pronunciations based on its learning. The performance improved further when trained with all 240 names. Boltzmann machine pronunciations generated for a few words are presented in Table 1.

Spell	Pron	110 wd train	240 wd train
Drew	d 9r u _	d 9r u _	d 9r u _
Teit	t _ ei t	t _ ei t	t _ ei t, g _ ei t
Vega	v E g &	v E g &, v E h &	v E g &, j E g &,v E h &
Wolf	w U l f	w U l f	w U l f

Table 1: Boltzmann machine output pronunciations

Final results on the entire database are not yet available because we have encountered problems with convergence of training on large datasets. These problems exposed the need to reformat some of the database such that it contains spellings aligned with their phonetic representations. Results on the entire training database will be presented at the conference.

6. CONCLUSION

Accurate pronunciation of proper nouns is a key issue in speech recognition applications. A major task in the training process of such a system is the design of the annealing schedule and selection of the number of units in the hidden layer. These parameters were found to significantly affect the performance of the network, and interact with various application-specific parameters such as the amount of training data, the size of the input shift register and the maximum error allowed in training. Optimization of these parameters is the subject of future research.

We also realized the need to align the spellings with the corresponding phonetic expression in our dictionary for ease of implementation. We have developed a dynamic programming algorithm that performs automatic alignment by introducing an *empty phoneme* at appropriate places and are currently updating the dictionary. We will provide a live demonstration with this modified system at the conference. All software and data will be made available at *http://www.isip.msstate.edu*.

7. ACKNOWLEDGEMENTS

We are grateful to the Systems and Information Sciences Laboratory of Texas Instruments, particularly Dr. Raja Rajasekaran, for providing the funding for this work. We also want to thank Dr. Barbara J. Wheatley for her insights into the Worldbet system, and help in the development of early versions of the training database.

REFERENCES

1. B. Wheatley and J. Picone, “Integrating Speech Technologies For Medical Applications,” presented at the Medical Applications of Voice Response Technology Conference in Pittsburgh, PA, Dec. 1989.
2. J. Picone, B.J. Wheatley and J. McDaniel, “On the Intelligibility of Text-To-Speech Synthesis of Surnames,” Texas Instruments Technical Report No. CSC-TR-91-002, pp. 1-34, Texas Instruments Inc., Dallas, TX, March 13, 1991.
3. G. Hinton and J. Anderson (Eds), *Parallel Models of Associative Memory*, Erlbaum Associates, NJ, 1981.
4. G. Hinton and T. Sejnowski, “Optimal Perceptual Inference”, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 448-453, Washington D.C., June 1986.
5. T.J. Sejnowski and C.R. Rosenberg, ““NETtalk: A Parallel Network That Learns To Read Aloud,” Tech. Rep. JHU/EECS-86/01, John Hopkins University, Baltimore, MD, 1986.
6. H.M. Meng, S. Seneff, and V.W. Zue, “Phonological Parsing for Reversible Letter-to-Sound/Sound-to-Letter Generation,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. II-1-II-4, Adelaide, Australia, April 1994.
7. J.J. Hopfield, “Neural Networks and Physical Systems with Emergent Collective Computational Abilities”, in *Proceedings of the National Academy of Sciences USA*, Vol. 79, pp. 2554-2558, 1982.
8. S. Kirkpatrick, C.D. Gellatt and M.P. Vecchi, “Optimization by Simulated Annealing”, in *Science*, Vol. 220, pp. 671-680, 1983.