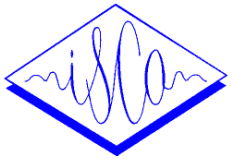

6th International Conference on Spoken Language Processing (ICSLP 2000)
Beijing, China
October 16-20, 2000

SPEECH ANALYSIS BY RULE EXTRACTION FROM TRAINED ARTIFICIAL NEURAL NETWORKS

Katrin Kirchhoff

Signal, Speech and Language Interpretation Laboratory
Department of Electrical Engineering
University of Washington
Seattle, USA
katrin@ee.washington.edu

ABSTRACT

A recent development in feature extraction is the use of neural network feature extractors, where the parameterized signal is passed through a neural network trained to discriminate between targets representing e.g. different phone classes or speakers. While the transformed feature representation often enhances class discriminability and thereby overall performance, the transformation performed by the network cannot directly be interpreted by human experts. However, explicit knowledge about this transformation could lead to the definition of a simpler function on the input features which might eventually be incorporated into the basic parameterization method. In this paper we investigate a rule extraction technique for transforming the trained network into a set of *if-then* rules capable of representing the transformation in a more transparent way, and apply it to the problem of distinguishing between the English fricative classes /f,v/ and /s,z/ from the TIMIT and OGI Numbers95 speech corpora.

1. INTRODUCTION

In recent years, the use of artificial neural networks (ANNs) as feature extractors for speech processing has increasingly gained popularity. The ANN approach to feature extraction usually consists of two steps: first, a basic preprocessing method (such as MFCC parameterization) is applied to the signal; second, an ANN is trained on this basic representation to discriminate between a set of classes (representing e.g. different speakers or phones). The result is a transformed feature representation generated by the ANN which arguably enhances class separability compared to the original feature set. The transformation performed by the ANN is essentially a non-linear discriminant analysis on the input feature space.

In [11] class-specific feature transformation ANNs were used together with linear affine transformations in a HMM recognition system for continuous digits, resulting in a 35% drop in word error rate. Fontaine et al. [4] trained a Multi-Layer-Perceptron (MLP) with two hidden layers on RASTA-PLP input features to discriminate between context-independent phones. After training, the outputs generated by the second hidden layer on the test set were used in place of the original features in a Gaussian mixture HMM system, which yielded a relative word error rate reduction of 25% on a large-vocabulary isolated word recognition task. Konig et al. [9] used the same type of MLP to maximize the discriminability between different speakers based on cepstral coefficients as input. The result was a 15% improvement over the baseline speaker identification system on the 1997 NIST speaker evaluation task. In [7, 8] sets of three-layer MLPs were used to map acoustic features to articulatory features; the latter were then employed both in isolation and in combination with acoustic features. While the combination method yielded a consistent improvement across different recognition tasks, acoustic conditions and different languages, articulatory features alone gave a significant improvement in highly noisy conditions on a continuous numbers recognition task. More recently, ANN feature transformations were applied to the Aurora task, a standard database for speech recognition over cellular networks, with improvements similar to the ones cited above [6, 13]. Related work was presented by Warakagoda and Johnsen [16], who showed that the different steps in the MFCC feature extraction procedure can be interpreted as the layers of a three-layer MLP and that the transformations at each layer can therefore be trained discriminatively.

Both recognition results and explicit feature selection experiments [8] support the assumption that ANN feature transformations supplement the original features by allowing information to be represented in a more class-discriminating way. However, there are also some disadvantages associated with this approach: first, the transformation learned during training is corpus-dependent; the networks can thus not readily be applied to new tasks [6]. This is unfortunate because of the additional effort associated with (re)training potentially large networks. It would be worthwhile to ascertain whether any systematic relationship exists between the same set of input features (e.g. MFCCs) and output classes (e.g. English monophones) that are invariant across different speakers, tasks, or acoustic conditions. This knowledge could then be used to define a simpler function (e.g. on subsets of features) which could in turn be integrated into the basic preprocessing method. The obvious difficulty is that ANNs are 'black boxes' which do not provide an explicit representation of the input-output mapping function learned during training.

In this paper, we therefore investigate a method to convert a feature transformation ANN into a set of Boolean *if-then* rules which are amenable to human inspection. After training, the ANN is first pruned and a rule search technique is subsequently applied to find valid combinations of input and hidden nodes resulting in activations of certain output nodes. We demonstrate this technique by applying it to an ANN trained to discriminate between the labio-

dental vs. coronal English fricatives /f,v/ and /s,z/. We present experimental results based on the TIMIT and OGI Numbers95 databases.

2. RULE EXTRACTION

The problem of interpreting trained ANNs has received much attention in the machine learning community but has so far not been applied to the kinds of speech feature transformation networks described above. Several different methods for converting ANNs into more explicit representations have been suggested, the most common of which is Boolean rule extraction (e.g. [5, 15, 2]), where the function relating input to output nodes in the ANN is described in terms of a set of logical if-then rules. These rules describe valid combinations of input nodes which result in the activation of an output node. The assumption underlying this technique is that the network to be analyzed uses activation functions which produce binary output values (0 or 1) for each node. As an example, consider a simple perceptron with a set of input nodes ($N1$ through $N4$), one output node ($N5$) and weights connecting the input to the output nodes. The output node has a threshold value, θ, associated with it. Let the activation function of node $N5$ be the sigmoid function

$$\sigma(x) = \frac{1}{(1 + e^{-a\,x})}$$

which maps x to a value close to 0 or 1. The 'sharpness' of this mapping can be controlled by the slope parameter a in the function denominator. The total activation O for node $N5$ in Figure 1 is computed by

$$O = \sigma(\sum_i w_i o_i - \theta)$$

where i ranges over all input nodes, w_i is the weight of the connection from node i to the output node, and o_i is the activation value of that node. Assume that the values of the input nodes are also binary, e.g. by virtue of them being the outputs from nodes in a previous layer. The goal is now to determine those combinations of input nodes which result in the activation of the output node. Since the input values are binary, we only need to look at the perceptron weights. The conditions under which $N5$ may be activated are given by the following rules (where $\vee$ = "or", $\wedge$ = "and", $\neg$ = "not"):

- if $N1 \vee N2 \wedge N4$ then $N5$
- if $N1 \wedge N2 \wedge \neg N3$ then $N5$
- if $N2 \wedge N4 \wedge \neg N3$ then $N5$, or, more compactly
- if $N2 \wedge (N4 \vee N1) \wedge \neg N3$ then $N5$

In a more complex network, e.g. a three-layer MLP, rule search proceeds from the output to the input layer. First, valid rules must be found for each output and hidden node. The combinations of hidden nodes can then be rewritten in terms of input nodes, which reveals relevant dimensions in the input space and they way in which they are combined to produce the desired outputs.

Following Fu [5], the search for rules for a given node can be organized into two steps: (a) find the minimal combinations of positive weights needed to exceed the node's threshold, and (b) add negative weights to each combination found to determine those weights which *must* be excluded. Fu uses a tree-based search,

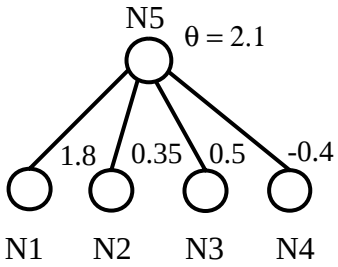

Figure 1: Example perceptron.

where, at each level in the tree, another weight is added to every weight set at the previous level. This procedure exhaustively checks all possible weight combinations; in our implementation we use a more efficient branch-and-bound technique for both step (a) and (b).

In order to further simplify the analysis process, several heuristics and 'shortcuts' can be used. First, the maximum number of possible rule antecedents can be limited to an upper bound k [5]. Second, descendants of a tree node need not be generated when the sum of positive weights in the current set minus all negative weights for the node in question exceeds the node's bias. The final rule sets can be condensed by subsuming more specific under more general rules and by introducing disjunctions (as in our example above).

Two problems with this technique quickly become obvious when applying it to feature transformation networks: first, the input feature are usually not binary but continuously valued. Second, the rule extraction algorithm is exponential in the number of nodes in a network. Since the size of our networks is often very large (on the order of tens of thousands of weights), this seems prohibitive. The first problem can be circumvented either by modifying the rule extraction algorithm to handle intervals rather than single values [14] or by quantizing the input features using a binary representation. In this paper we pursue the quantization approach. The complexity problem can be addressed by pruning the network before proceeding with rule extraction. Quantization and pruning are described in Sections 4 and 5.

3. DATABASE

The task we consider in this paper is to discriminate between two classes of English fricatives, the labiodental fricatives /f,v/ vs. the coronal fricatives /s,z/. For speech data we use fricatives extracted from two different corpora, TIMIT and the OGI Numbers95 corpus. The latter consists of telephone-speech continuous numbers. These corpora were chosen specifically to investigate how the distinction between the place of articulation of these fricatives is encoded in clean vs. telephone speech. In each case, instances of the relevant fricatives were extracted from the signal with two frames of left and right context, based on the hand-labelled transcriptions distributed with the corpora. Examples were then relabelled as either one of the two classes and a separate MLP was trained for each corpus. In both cases, the same number of training (10000) and cross-validation samples (1300) were used. For

testing, all relevant fricative examples from the respective test sets (TIMIT core test set and Numbers95 development set) were used, resulting in 25542 test samples for TIMIT and 246125 samples for Numbers95. In each case, the input representation consisted of 12 MFCC features plus energy and first derivatives. The MLPs used a window of 5 input frames and 75 hidden units each. Training was done using the relative entropy between the output and the target distributions as the objective function. Both the hidden and the output layer used the sigmoid activation function.

4. INPUT FEATURE QUANTIZATION

As mentioned above, the continuous input features need to be converted into a binary representation in order to fulfill the assumptions underlying the rule extraction algorithm. In addition to this, quantization in itself can be a 'speech mining' tool; depending on the quantization method, it can sometimes highlight important intervals in the data values.

We investigated three different quantization schemes: equal interval binning, mean splitting and a minimum description length quantization algorithm [3]. The first method divides the range of values for a given feature into equal intervals and assigns 1 to the bin which a given feature value falls into and 0 to all others. The second method first normalizes the features to zero mean and unity variance and then splits along the mean of the distribution: if a given feature value is larger than or equal to the mean, 1 is assigned, else zero. Minimum description length quantization takes into account not just the feature values but also the class labels associated with them and recursively partitions the data into two subsets so as to minimize the class entropy. The class entropy for the binary partition of the set S of N samples for feature F that is induced by the boundary point T is defined as

$$H(F,T;S) = \frac{|N_1|}{|N|}H(S1) + \frac{|N_2|}{|N|}H(S_2)$$

where N_1 and N_2 are the number of samples in subsets S_1 and S_2 respectively. The splitting process stops when the following criterion is met:

$$Gain(F,T:S) < \frac{log_2(N-1)}{N} + \frac{\Delta(F,T,;S)}{N}$$

where

$$\Delta(F,T;S) = log_2(3^k - 2) - [k*H(S) - k_1*H(S_1) - k_2*H(S_2)]$$

and k_i is the number of class labels represented in set i. The resulting bins are then used as in equal interval quantization. The goal is to create partitions that are as "pure" as possible, i.e. that ideally contain examples of one class only.

We found that both binning methods decreased accuracy drastically, probably by enlarging the input feature space from 26 to 200-300 features and thereby increasing the number of parameters in the network. Mean splitting worked surprisingly well, resulting in frame-level accuracy rate reductions (Table 4) that are significant but not overly large.

5. NETWORK PRUNING

In order to limit the complexity of the rule search algorithm (and the size of the rule set generated by it), it is advisable to first prune the trained ANN by removing superfluous nodes and weights. For the experiments reported in this paper, we used a brute-force weight elimination procedure, which iteratively deletes a single weight from the network, performs a forward pass with the reduced network and and computes the changes in the cross-validation accuracy. If the accuracy is reduced by more than some amount δ, the weight is restored, else it is eliminated entirely from the network. Figure 2 shows the reduction in % of the number of parameters versus the reduction in frame-level classification accuracy on the test set, for $\delta = 0.0$. We see that the network size can be reduced considerably with only small losses in test accuracy (Figure 2). This pruning method is computationally very expensive. We are therefore currently investigating more efficient, theoretically well-defined pruning approaches, such as training with weight-decay [12].

Feature type	TIMIT	Numbers95
continuous	95.51%	90.59%
quantized	94.83%	86.94%

Table 1: Frame-level accuracy rates for fricative classification (/f,v/ vs. /s,z/) using continuous and quantized input features.

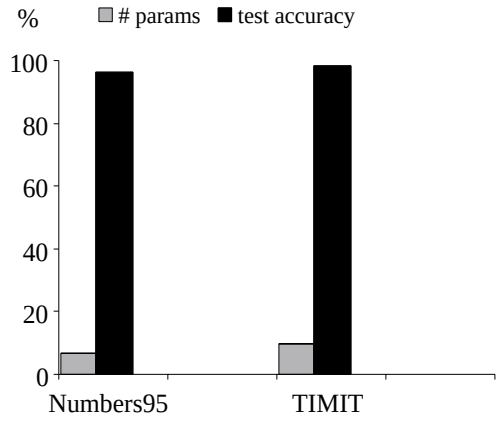

Figure 2: Effect of pruning: reduction in % of the number of parameters (light boxes) vs. reduction in % in test set classification accuracy (dark boxes) compared to the original, unpruned network.

6. EXPERIMENTS

The rule extraction program was run on the pruned networks trained on quantized input features. We observed during pruning that the largest reductions in CV accuracy were correlated with the magnitude of the weights; for this reason, we restricted rule search to be applied only to the largest positive and negative weights, excluding those weights which were at least an order of magnitude smaller. For each category and each corpus we obtained between 10 and 20 rules (in compact notation), specifying several hundred conditions when compiled out into individual rules. Space prevents us from exhaustively listing all rules; several examples of the rules are shown below. Fi stands for frame i in the input win-

dow and Cj is the j'th cepstral coefficient (where C1 through C12 are the basic cepstral coefficients, C13 is energy, C14-C26 are the deltas).

/f,v/ (Numbers95):
if $(F2C1 \wedge F5C1 \wedge F5C13) \wedge \neg(F1C4 \wedge F1C5 \wedge F1C13 \wedge F1C25 \wedge F2C25 \wedge F5C3)$ then /f,v/
/s,z/ (Numbers95):
if $(F3C14 \vee F4C13) \wedge F3C17 \wedge F4C14 \wedge F5C13 \wedge F5C26 \wedge \neg(F1C1, F2C2, F2C3 \wedge F2C4 \wedge F2C13 \wedge F3C1 \wedge F1C1) \wedge \neg(F2C1 \wedge F5C1)$ then /s,z/
/f,v/ (TIMIT)
i $(F5C15 \wedge F1C1 \wedge (F4C26 \vee F2C26)) \wedge \neg(F1C14 \wedge F1C2 \wedge F5C13 \wedge F5C3)$ then /f,v/
/s,z/ (TIMIT)
if $F1C26 \wedge F1C2 \wedge (F1C15 \vee F2C2) \wedge \neg(F1C14 \wedge F1C16 \wedge F2C1 \wedge F2C3 \wedge F3C1 \wedge F4C1 \wedge F5C1 \wedge F4C3 \wedge F4C14) \wedge F5C17$ then /s,z/

Rules like these are still fairly complicated to read. What is of interest is mainly whether any significant overlap exists between the rule sets for the different corpora, as well as notable differences. From the overall analysis of our rule sets it appears that the input features $F1C4 \wedge F2C4 \wedge \neg F1C14$ are are relevant for identifying /f,v/ in both ANNs. The Numbers95 network additionally makes use of $F5C12 \wedge F5C25 \wedge C5C26 \wedge \neg(F5C13 \wedge F5C15 \wedge F5C17)$ whereas classification in the TIMIT network involves $\neg(F2C13 \wedge F2C26)$
$\wedge F2C1 \wedge F3C1 \wedge F4C1 \wedge F5C1$. For the /s,z/ class both networks frequently use $F1C13 \wedge F1C26 \wedge F5C13 \wedge F5C26$. Additionally, for the Numbers95 corpus $F1C1 \wedge F5C1 \wedge \neg(F1C5 \wedge F1C8 \wedge F1C13 \wedge F1C16)$ seem to be relevant; for the TIMIT task, $F1C14 \wedge \neg(F5C26 \wedge F1C4 \wedge F2C4)$ are used. It is as yet unclear how universal these rules are; further experiments on clean vs. telephone speech corpora need to be carried out in order to determine whether our observations generalize to other data sets.

7. CONCLUSIONS

We have presented a technique for converting ANNs for non-linear discriminant analysis of speech features into a more explicit and interpretable representation, viz. logical *if-then* rules. These rules are not meant to replace ANNs as classification devices; they are intended to provide some insight into the nature of the feature transformation distinguishing certain phonetic classes from others. This method can be used as a general speech analysis tool to uncover how speech information is encoded in parameterized speech signals. As such, it can complement and enhance other speech analysis methods e.g. those which look at the mutual information between input features and class labels [10, 1, 17] but do not consider the way in which features are combined to encode certain categories. We are currently looking at ways of simplifying pruning, particularly by weight decay training, and at other heuristics to constrain rule search. A future application of this speech analysis techniques might be a comparison of rules extracted under different acoustic conditions, e.g. clean, noisy and reverberant speech.

Acknowledgements
We acknowledge the use of the ICSI Quicknet software for training the baseline ANNs.

REFERENCES

[1] J.A. Bilmes. Maximum mutual information based reduction strategies for cross correlation based joint distributional modeling. In *Proceedings of ICASSP*, 1998.

[2] M.W. Craven and J.W. Shavlik. Using sampling and queries to extract rules from trained neural networks. In *Proceedings of the Eleventh International Conference on Machine Learning*, 1994.

[3] U.M. Fayyad and K.B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of IJCAI*, pages 1022–1027, 1993.

[4] V. Fontaine, C. Ris, and J.M.Boite. Nonlinear discriminant analysis for improved speech recognition. In *Proceedings of Eurospeech*, pages 2071–2074, 1997.

[5] L. Fu. Rule generation from neural networks. *IEEE Transactions on Systems, Man and Cybernetics*, 24:1114–1124, 1994.

[6] H. Hermansky, D.P.W Ellis, and S. Sharma. Tandem connectionist feature extraction for conventional HMM systems. In *Proceedings of ICASSP*, 2000.

[7] K. Kirchhoff. Combining articulatory and acoustic information for speech recognition in noisy and reverberant environments. In *Proceedings ICSLP*, pages 891–894, 1998.

[8] K. Kirchhoff, G. Fink, and G. Sagerer. Conversational speech recognition using acoustic and articulatory input. In *Proceedings of ICASSP*, 2000.

[9] Y. Konig, L. Heck, M. Weintraub, and K. Sonmez. Nonlinear discriminant feature extraction for robust text-independent speaker recognition. In *Proceedings of the ESCA Workshop on Speaker Recognition, Avignon, France*, 1998.

[10] A. Morris. An information theoretical investigation into the distribution of phonetic information across the auditory spectrogram. *Xomputer Speech and Language*, 2:121–136, 1993.

[11] M. Rahim, Y. Bengio, and Y. LeCunn. Discriminative feature and model design for automatic speech recognition. In *Proceedings of Eurospeech*, pages 75–78, 1997.

[12] R. Setiono. A penalty-function approach for pruning feedforward neural networks. *Neural Computation*, 9:185–204, 1997.

[13] S. Sharma, D. Ellis, S. Kajarekar, P. Jain, and H. Hermansky. Feature extraction using non-linear feature transformation for robust speech recognition on the Aurora database. In *Proceedings of ICASSP*, 2000.

[14] S. Thrun. Extracting provably correct rules from artificial neural networks. Technical Report IAI-TR-93-5, University of Bonn, 1993.

[15] G. Towell and J. Shavlik. The extraction of refined rules from knowledge based neural networks. *Machine Learning*, 131:71–101, 1993.

[16] N.D. Warakagoda and M.H. Johnsen. Neural network based optimal feature extraction for ASR. In *Proceedings of Eurospeech*, 1999.

[17] H.H. Yang, S. van Vuuren, S. Sharma, and H. Hermansky. Relevance of time-frequencey features for phonetic and speaker-channel classification. *Speech Communication*, 2000.