TR-I-0090

Parallelism, Hierarchy, Scaling in Time-Delay Neural Networks for Spotting Phonemes and CV-Syllables

Hidefumi Sawai, Alex Waibel, Patrick Haffner,
Masanori Miyatake and Kiyohiro Shikano

1989. 7.25.

Abstract

Syllable or phoneme spotting, if reliably achieved, provides a good solution to the spoken word and/or continuous speech recognition problem. We previously showed that the Time-Delay Neural Network (TDNN) provided excellent recognition performance for all phonemic subcategories (nasals, fricatives, vowels, etc.). To extend this encouraging performance of TDNNs to all phoneme recognition and word/continuous speech recognition, we show several techniques: Firstly, we show that it is indeed possible to scale up the TDNN to a large phonemic TDNN aimed at discriminating all phonemes without loss of recognition performance and without excessive training tokens. Secondly, we propose fast back-propagation learning methods which make it possible to train a large phonemic TDNN within 1.5 hours. Finally, we show several methods for spotting Japanese CV syllables/phonemes in input speech based on TDNNs: we constructed a TDNN which can discriminate a single CV-syllable or phoneme. Syllable and phoneme spotting experiments show excellent results, including syllable and phoneme spotting rates of better than 96.7% and 92% correct, respectively. These spotting techniques are proved to be a good step toward continuous speech recognition.

ATR 自動翻訳電話研究所
ATR Interpreting Telephony Research Laboratories

Contents

1. Introduction

Connectionist models are now applied to many tasks in speech recognition. However, there are many obstacles to overcome before it will be feasible to extend the use of the models from restricted speech recognition tasks to continuous speech recognition systems. Scaling is one such obstacle. Scaling a connectionist model to a larger connectionist system is difficult, because larger networks require increasing amounts of training time and data. Thus, the complexity of the optimization task quickly reaches computationally unmanageable proportions. In this paper, we propose several techniques that allow us to "grow" larger nets in an incremental and modular fashion without the need for excessive training time and additional data. The training of networks was performed using the back-propagation procedure[1].

The large amount of training time is another problem for network learning. The back-propagation learning algorithm has proven its ability to make a layered neural network compute any kind of complex decision surface[1]. However, these high performance results were attained at the expense of training speed, which could be a major obstacle to training a large kowledge base. We show in Section 2, that it is possible to increase this speed by several orders of magnitude, while maintaining the same recognition performance. Moreover, our learning algorithm enables us to tackle very large tasks and increase the limit size of phonemic tasks which neural networks can be trained to handle in computational comfort.

Spotting CV-syllables is a good approach to word and continuous speech recognition in Japanese because there are only about one hundred syllables. An earlier work was done with only 7 kinds of English demi-syllables[4]. The architecture of the TDNN[2,3] is well suited to spotting CV-syllables/phonemes because the shift-invariant structure makes it possible to correctly spot them even in the neighboring positions of syllable/phoneme tokens. If we train a neural network which can reliably discriminate one syllable, and also prepare all kinds of neural networks "in parallel", the spotting of any syllable can, in principle, be achieved for any input utterances. However, because there are about one hundred syllables which can be chosen, one significant problem will occur when we choose as training tokens all possible syllables except the specific syllable to be discriminated. As the first step in spotting Japanese CV-syllables, we arbitrarily chose the syllable "BA" as a typical Japanese syllable. As training tokens, the "non-BA" syllables "DA", "GA", "PA", "TA" and "KA" are chosen because they might be confused with "BA". We might be able to automatically discriminate all possible syllables other than the five syllables used as training tokens. If this is true, both training time and the number of tokens will be greatly reduced.

In general, spotting phonemes is also an effective approach to speech recognition in other languages. We study the phoneme spotting approach for comparison with the syllable spotting approach described above. We examined two approaches to spotting phonemes: one is a hierarchical decision approach and the other is an instantaneous whole phoneme spotting approach.

In the former case, phoneme group networks which can discriminate one phoneme group (ex.:"BDG") are trained. "BDG", "PTK", "MNsN", "SShHZ", "ChTs", "RWY" and "AIUEO" are chosen as phoneme groups. As well as training phoneme group networks, the corresponding "intra-group" networks are also provided with training tokens for each subcategory (ex.:"B", "D" and "G" for the "BDG" intra-group network). Spotting experiments are performed by determining the phoneme category which is involved in the phoneme group. This "hierarchical" approach to spotting phonemes is critical if an incorrect phoneme group were determined. However, it is advantageous that only 7 phoneme groups and 7 intra-group networks need be prepared rather than about one hundred CV-syllable networks.

For the latter case, scaling from modular networks to a larger network is a critical problem. Spotting all major Japanese phonemes (i.e., "B, D, G, P, T, K, M, N, sN, S, Sh, H, Z, Ch, Ts, R, W, Y, A, I, U, E, O") was performed at the same time using a large phonemic TDNN scaled up from modular TDNNs. This scaling technique will be introduced in Section 5.2. Training TDNNs was performed by the back-propagation procedure with the fast algorithm we developed[6]. In the final section, we will compare the performance of the three approaches to CV-syllable spotting and phoneme spotting.

2.Training with Fast Back-Propagation Methods

In this section, we propose a fast learning algorithm for the back-propagation procedure[6] in phoneme recognition and spotting.

2.1. The Problem of Training Time in Scaling Neural Networks

Several techniques for neural network training exist[1,5]. For example, for a network aimed at the discrimination of the voiced stops (a BDG-net), approximately 6,000 connections had to be trained over about 800 training tokens. An identical net can achieve discrimination among the voiceless stops ("P", "T" and "K"). To extend our networks to the recognition of *all* stops, i.e., the voiced *and* the unvoiced stops (B,D,G,P,T,K), a larger net is required. However, it took inordinate amounts of learning to arrive at the trained net (several weeks on a 4-processor Alliant!). Although going from voiced stops to all stops is only a modest increase in task size, about 18,000 connections had to be trained. To make matters worse, not only does the number of connections have to be increased with task size but, in general, the amount of training data required for good generalization of a larger net has to be increased as well. Naturally, there are practical limits to the size of a training database and more training data translates into even more learning time. Learning is further complicated by the increased complexity of the higher dimensional weightspace in large nets as well

as the limited precision of our simulators. Therefore, we have to develop fast learning methods for the back-propagation procedure.

2.2. Improving Training Speed

The goal is to reduce the error measure to zero within the smallest learning time. As a function of the connection weights, this error function defines a complex surface, and learning may be seen as a trajectory on this surface, moving down along the steepest slope, preferably toward a global minimum. There are several ways to accelerate convergence:

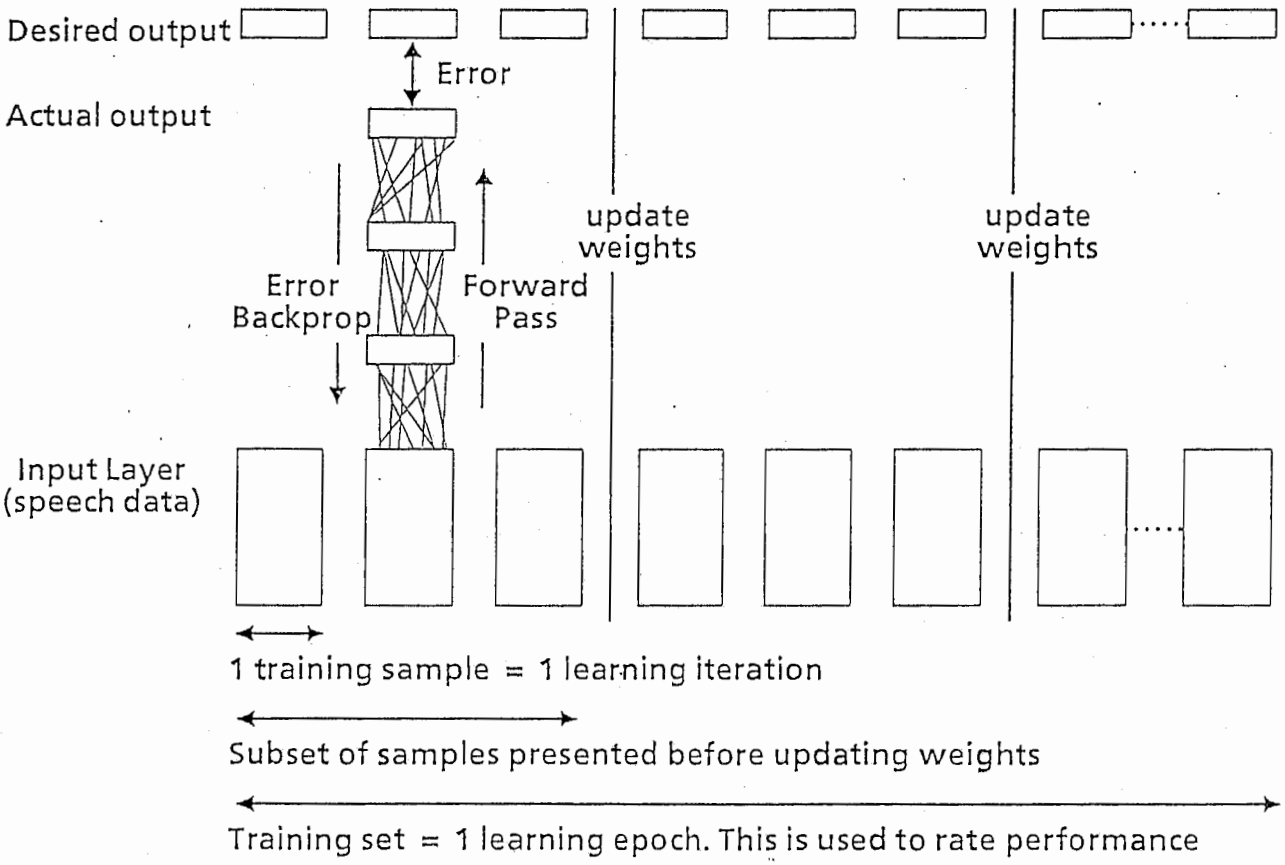

Fig.1. Learning the trainig set

(1) Model a steeper error surface without flat spots, which may be done by the appropriate choice of the sigmoid function or output error measure.

(2) Define an appropriate leaning strategy: in which order to present patterns, when to update the connection weights (as shown in Fig.1).

(3) Carefully choose the parameters presented as the gradient step size, the momentum and the weight decay[9]. These parameters will be scaled to:

- Make the weight trajectory as straight as possible.
- Minimize oscillations.
- Achieve a fast but controlled learning speed.
- Attain good generalization on test data.

2.3. Methodology

2.3.1. The Sigmoid Function

The back-propagation learning rate is proportional to the values of the sigmoid function f and its derivative f'. As seen in Fig.2, these functions flatten out at infinity, leading to a very slow learning rate. A simple way to prevent this is to add some small constants to f or f'.

2.3.2. Scaling the Step Size

The optimal value of the step size may vary widely with time, which is consistent with the large variations of slope and curvature on the energy surface. Thus, setting the step size is one of the most difficult problems with back-propagation. The previous literature[11,15] proposed different algorithms to scale the step size. However, they seem difficult to tune. Furthermore, as they

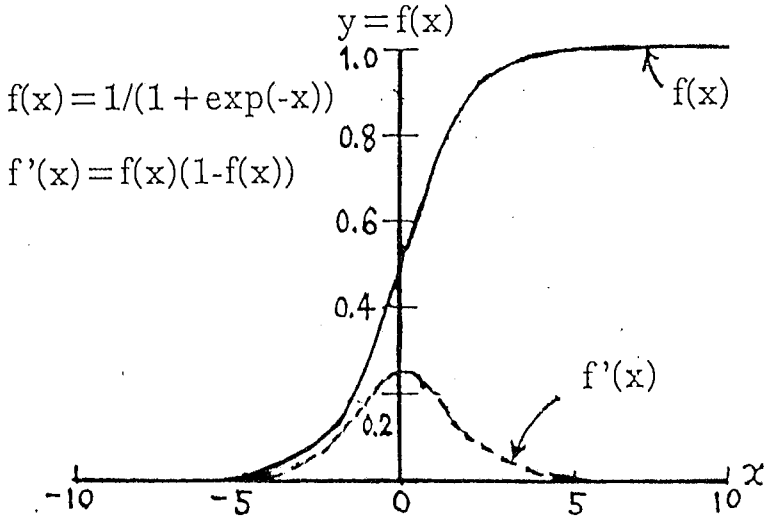

Fig.2. The sigmoid function and its derivative.

generally try to scale the step size to its maximum value, they lead to large jumps in the weight space which are difficult to control.

While these algorithms do lead to significant improvements in speed, they are very sensitive to overshooting. We have added an overshooting control that, at each updating iteration, limits the norm of the vector $\varepsilon \cdot \text{grad}(E)$ to a fixed value ω (where ε is a step size, and E is an error).

2.3.3.The Weight Updating Frequency

Splitting a large and often highly redundant training set into smaller subsets for the purpose of weight updating may be very advantageous. At the beginning of the learning phase, one subset is enough data for a network which is only acting as a rough classifier. As a consequence, updating weights over any subset may be as effective as updating weights over the whole training set at the beginning of training. At the end of the learning phase, fine-grained leaning and a large training set are needed. However, the difference between two consecutive leaning subsets may be considered noise which prevents local minima.

2.3.4. Skipping Samples

When learning a large sample database, the learning program tends to spend most of its time trying to learn a small number of unclassified samples, while most of the other samples already yield an output error close to zero. It is then a waste of time to perform both forward and backward passes on these learned samples. To prevent this, one has only to set a minimum error: if a sample output error is below this minimum, no backward pass is performed. This algorithm may be improved for samples whose error is much below this minimum. With this methods, we commonly skip 75% of the training samples at the end of the learning phase, and save the same percentage of CPU time per Epoch.

2.4. Our Learning Procedure

Among all the possible improvements presented in the previous sections, we have selected those giving the largest increase in performance, both in terms of learning speed and generalization capacity:

(1) The standard sigmoid is used.

(2) The input activations are normalized to range from -1 to +1.

(3) As our tasks have a large number of classes, McClelland's New Error is used.

(4) The weights are updated according to the weight updating procedure: the initial updating period is 9 iterations, and is incremented by 3 iterations at each Epoch, until it reachs 72 iterations (as shown in Fig.1).

(5) Training samples are skipped when their output error is below 0.001. The maximum number of Epochs during which they can be skipped is 5.

(6) The step size is limited by the overshooting control procedure. When it is not limited, the step size is 0.01.

(7) The momentum is scaled by the procedure, with a minimum value of 0.5 and a maximum value of 0.99.

3. Parallelism in Spotting Japanese CV-Syllables

In this section, we describe a technique to spot Japanese CV-syllables using the TDNN.

3.1. Architecture

The architecture of a TDNN for spotting a Japanese CV-syllable is shown in Fig.3. If the TDNNs in Fig.3 can be prepared for all syllables in parallel, spotting any syllable becomes possible. The input layer of the network is the same as a "BDG" network for discriminating "B", "D" and "G"[2]. The first hidden layer is composed of 4 units compared to 8 units for the "BDG" network. This is because the number of categories is reduced to two (i.e., one specific syllable and others). The second hidden layer is also composed of two units corresponding to two outputs in the output layer. The TDNN has 241 inputs, 2 outputs, 313 total units and 2,946 connections. As in phoneme recognition by the "BDG" network, 16 melscale spectrum coefficients serve as inputs to the network. Input speech, sampled at 12 kHz, was Hamming-windowed and a 256-point FFT computed every 5 msec. Melscale coefficients were computed from the power spectrum[3] and adjacent coefficients in time collapsed, resulting in an overall frame rate of 10 msec . The coefficients of the input token (in this case 15 frames of speech centered around the hand labeled vowel onset) were then normalized to fall between -1.0 and +1.0, with the average at 0.0. Fig.1 shows the resulting coefficients for the speech token "BA" as input to the network, with positive values shown in black squares and negative values in grey squares.

3.2. Experiment Conditions and Database

Experiments in spotting a Japanese CV-syllable are carried out by scanning the corresponding TDNN among input word speech uttered by one male speaker. Spotting syllables in words is an important step toward continuous speech recognition. If the spotting experiments are successful, this work even has the potential to spot any syllable in continuous speech.

For the evaluation of spotting Japanese CV-syllables, a large database of 5,240 common Japanese words were uttered in isolation by a native Japanese speaker. All utterances were recorded in a soundproof booth and digitized at a 12 kHz sampling rate. The database was then split into a training set and a testing set of 2,620 utterances each, from which the actual syllabic tokens were extracted. For the CV-syllable network training, tokens from various contexts of syllables containing "BA", "DA", "GA", "PA", "TA" and "KA" were selected. Both

training and testing tokens were analyzed and windowed by a 256-point Hamming window, and then converted to feature parameters represented as 16th melscale FFT outputs every 10 msec.

3.3. Training in the TDNN for Spotting CV-Syllables

Training of our TDNN was done by a fast program of the back-propagation procedure described in Section 2. Although there are about one hundred "non-

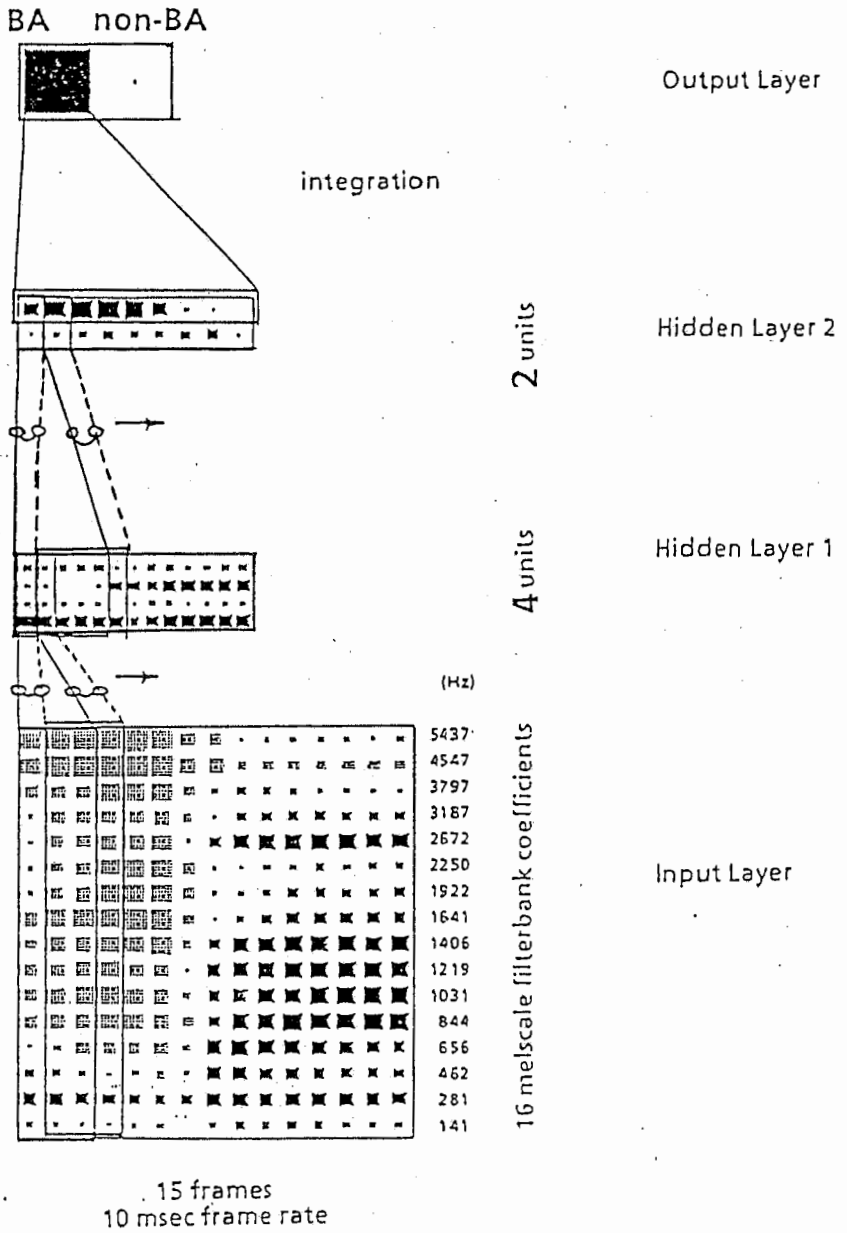

Fig.3. TDNN for spotting a Japanese CV-syllable

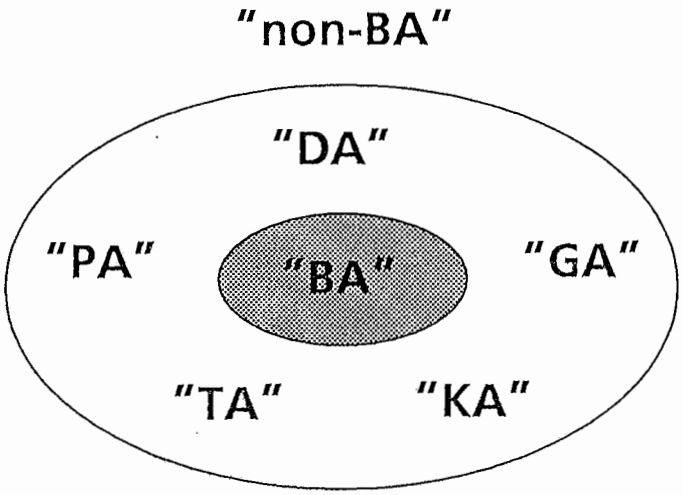

Fig.4. Training syllable tokens confusable with "BA" in Japanese

BA" syllables, it is inefficient to use all of them. Thus, only five "non-BA" syllables such as "DA", "GA", "PA", "TA" and "KA" are used, because these syllables are particularly confusable with "BA". The decision hyperplanes for discriminating between "BA" and "non-BA" may be formed without excessive training tokens for the "non-BA" syllables shown in Fig.4. Actually, as training tokens for "BA" syllables, we used 53 words with one or more "BA" syllables chosen from the even-numbered words of the 5,240 common Japanese word database. Similarly, as training tokens for "non-BA", 752 tokens which contain the syllables "DA", "GA", "PA", "TA" and "KA" were used. The boundaries between the consonants and the following vowels in the training syllables are centered at the input layer in the TDNN. Training took only several minutes using our fast program of the back-propagation procedure on an 8-processor Alliant super-minicomputer!

3.4. Results of Spotting CV-Syllables

For testing CV-syllable spotting, 61 words containing one "BA" syllable each were taken from the testing word set. 138 "non-BA" CV-syllables (not only "DA",

Table.1. Results of Spotting CV-syllables (ex.: "BA")

Syllable	"BA"	"non-BA"	Total
#of syllables	61	138	199
#correct/total	59/61	137/138	196/199
Recognition rate(%)	96.7	99.3	98.5

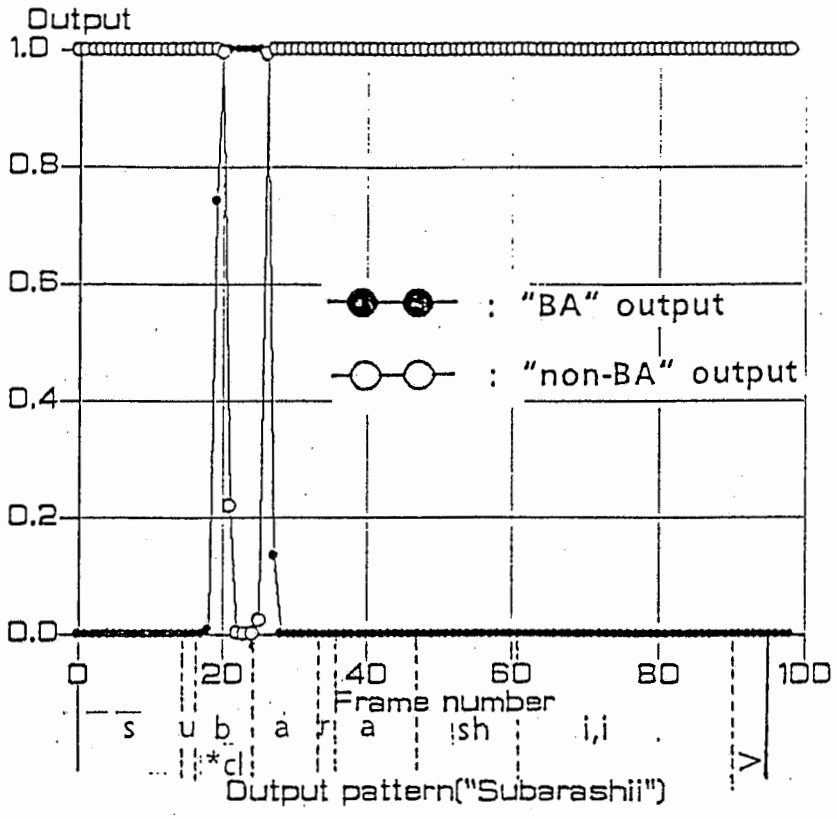

Fig.5. An example of CV-syllable spotting results: input utterance is "SUBARASHII".

"GA", "PA", "TA" and "KA" but also other possible CV-syllables) were included in the 61 words. Given an unknown input CV-syllable, "BA" and "non-BA" were recognized according to the following criteria:

if o("BA") > o("non-BA"),

then the input CV-syllable is "BA",

if o("non-BA") > o("BA"),

then the input CV-syllable is "non-BA".

Where, o("BA") and o("non-BA") are the activation values of two outputs corresponding to "BA" and "non-BA", respectively.

In Table 1, spotting performance is shown. The syllable spotting rate is defined as the rate of correct output decisions of the network. The network achieved a "BA" spotting rate of 96.7% and performed well in inhibiting all possible CV-syllables (not only "DA", "GA", "PA", "TA" and "KA" but also all other syllables, including boundary positions) belonging to the "non-BA" category at a rate of 99.3%.

Fig.5 shows an example of spotting results, where the input utterance is the Japanese word "SUBARASHII". The output of "BA" is robustly fired at the

position of "BA", and all other syllables are well inhibited as shown in the "non-BA" output values.

Spotting errors occured at the beginning of the word "ASHIBA" where "A" misfired as "BA", and in the two intermediate parts of words such as "KEIBATSU" and "SHIBARAKU" where the "BA" outputs were not robustly fired.

4. Hierarchy in Spotting Phonemes

In this section, we propose one of the spotting techniques for all Japanese phonemes: a hierarchical spotting technique.

4.1. Hierarchical Architecture

For a good comparison with the CV-syllable spotting approach described in Section 3, we examined the phoneme spotting approach with the hierarchical decision criterion shown in Fig.6. This criterion is based on first spotting phoneme groups, and then spotting phonemes determined by the results of the phoneme group spotting.

For the phoneme group TDNNs, networks similar to the one shown in Fig.3 were constructed with a slight modification: 8 units were used in the first hidden layer instead of the 4 units shown in Fig.3, because phoneme group TDNNs need a greater hidden unit capacity than the syllabic TDNN. All phonemic categories are divided into 7 phoneme groups, i.e., "BDG", "PTK", "MNsN", "SShHZ", "ChTs", "RWY" and "AIUEO". The overall network has 241 inputs, 37 outputs, 2,119 total units and 88,275 connections.

For testing phoneme spotting, a "hierarchical" decision was also made by first determining one or more phoneme groups and after that, by discriminating phonemes in the phoneme group determined.

4.2. Experiment Conditions and Database

As with the experiment conditions of the CV-syllable spotting described in Section 3.2, the even-numbered Japanese common words were used as training tokens. These tokens were analyzed with the same experiment conditions, and then all phonemic tokens were extracted from a variety of training word phoneme contexts by inspection.

Testing on spotting phonemes was performed for 50 words of the phoneme balanced words selected from the Japanese database and additional words, so that all phonemes would appear equally often.

4.3. Training in the TDNN for Spotting Phonemes

As with the CV-syllable spotting networks, the phoneme group networks were trained by our fast back-propagation procedure using both the corresponding training tokens in a phoneme group (ex.:"BDG") and the residual phoneme group

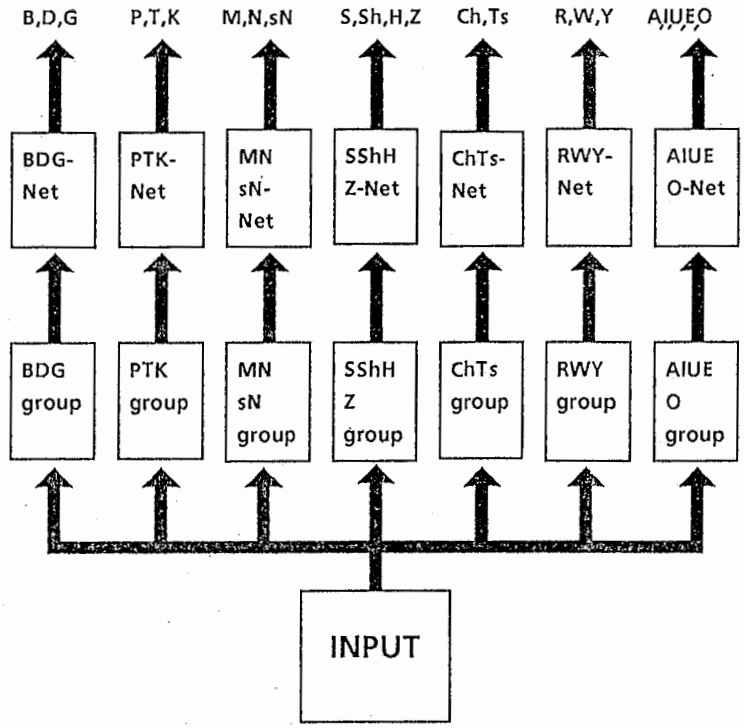

Fig.6. A Hierarchecal structure of TDNNs for spotting phonemes

tokens (ex.: tokens other than "BDG"). The residual training tokens were selected by easily confused phonemic group tokens based on the phoneme group recognition confusion matrix. This selection of residual training tokens is necessary to reduce the number of tokens and time for phoneme group network training.

These 7 phoneme group TDNNs were trained by our fast back-propagation procedure. The 7 intra-group TDNNs[7] were also trained to discriminate phonemes, for example, between "B", "D", and "G" corresponding to the "BDG" phoneme group TDNN. For the intra-group network training (ex.:"BDG" net), up to 600 training tokens per phoneme category were used[3]. These intra-group networks consist of "BDG", "PTK", "MNsN", "SShHZ", "ChTs", "RWY" and "AIUEO". Finally, the 7 kinds of phoneme group networks and intra-group networks trained are constructed in a hierarchical fashion as shown in Fig.6. Training took about 1.5 hours for the overall network.

4.4. Results of Spotting Phonemes

Typical spotting results are shown in Fig.7, where the Japanese word "TORIATSUKAU" was spotted. In Fig.7, the input layer is shown at the bottom, the output activation pattern of the phoneme group networks is shown in the middle, and the final output activation pattern of the intra-group networks is shown at the top. The word utterance is manually labeled by inspection to check coincidences between labels and spotting results.

In the Fig.7 phoneme group spotting results, although some misfired

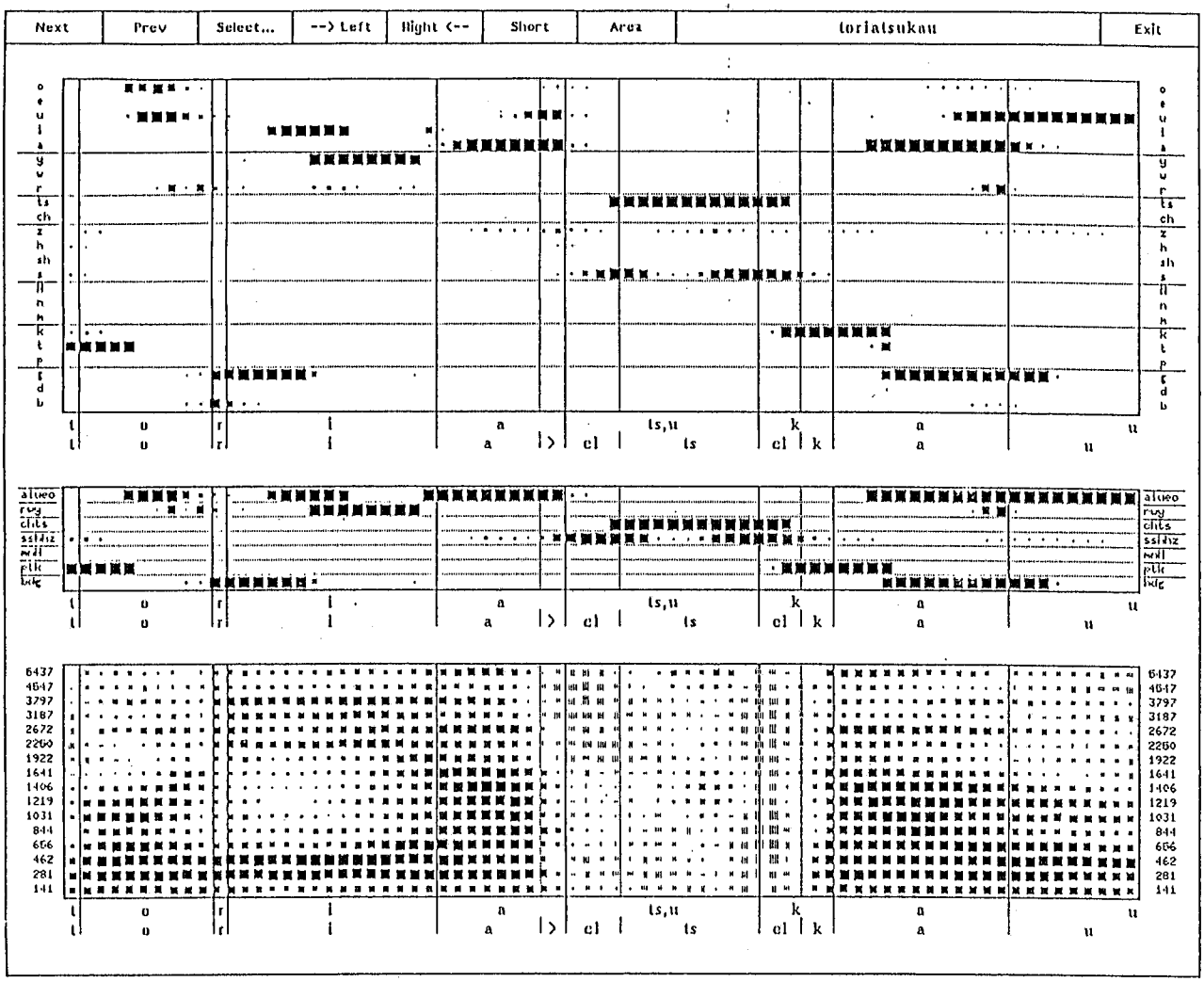

Fig.7. An example of phoneme spotting by a hierarchical TDNN: input utterance is "TORIATSUKAU".

activation patterns such as "BDG", "SShHZ", "RWY" are observed, correct spotting results were obtained for all input phonemes except the devocalized "U" after "TS". The spotting rate of correct phoneme groups is better than 95%, averaged over the 50 testing words.

In the final phoneme spotting results (i.e.,intra-group spotting), activation outputs belonging to correct phoneme categories are also observed in the phoneme group spotting results. The spotting rate of correct phonemes is 92%, averaged over the same 50 testing words. The insertion error equalled the number of phonemes because the networks were trained only from training tokens extracted from phoneme center positions. (Table 4 summarizes the results). Nonetheless, the results of the phoneme spotting are excellent because of the *shift invariant* property in TDNN.

5. Scaling in Spotting Phonemes

5.1. Experiment Conditions and Database

For experiments on phoneme recognition, we have used a large vocabulary database of 5,240 common Japanese words[10]. Training tokens were extracted from the even numbered word phonemes, and up to 600 tokens per phoneme were randomized within each phoneme class. For performance evaluation, we have run all experiments on the testing tokens only, as extracted from the odd numbered word phonemes, i.e., on tokens *not* included during training.

For performance evaluation of phoneme spotting, we have used the same 50 phoneme balanced words used in the hierarchical spotting experiment.

5.2. Scaling Modular TDNNs to Larger Phonemic TDNNs

We previously showed that as a strategy for efficient construction of larger networks we have found the following concepts to be extremely effective: *modular, incremental learning, class distinctive learning, connectionist glue, partial and selective learning, and all-net fine tuning*[7]. These techniques were applied to the tasks of recognizing all consonants and phonemes in our database. In the following we describe our attempts at building larger nets, and note that numerous alternative solutions remain to be explored.

5.2.1. Consonant Network Architecture

Our consonant TDNN (shown in Fig.8) was constructed modularly from networks aimed at the consonant subcategories, i.e., the BDG-, PTK-, MNsN-, SShHZ-, TsCh- and the RWY-tasks. Each of these nets had been trained before to discriminate between the consonants within each class. Hidden layers 1 and 2 were then extracted from these nets, i.e. their weights copied and frozen in a new combined consonant TDNN. In addition, an interclass discrimination net that distinguishes between the consonant subclasses was trained. The structure of this network was very similar to other subcategory TDNNs, except that we have allowed for 20 units in hidden layer 1 and 6 units (one for each coarse consonant class) in hidden layer 2. The weights leading into hidden layers 1 and 2 were then also copied from this interclass discrimination net into the consonant network and frozen. Three connections were then established to each of the 18 consonant output categories (B,D,G,P,T,K, M,N,sN,S,Sh,H,Z,Ch,Ts,R,W and Y). The overall network architecture is illustrated in Fig.8. The all consonant TDNN has 241 inputs, 18 outputs, 1,359 total units and 55,754 connections. All free weights were initialized with small random weights and then trained by the back-propagation learning procedure. After training all free weights, all weights (including fixed weights between hidden layers 1 and 2, and the input layer) were free and fine-tuned. Training took only about one hour with our fast learning methods using 200 tokens per each phoneme category (total tokens =3,600).

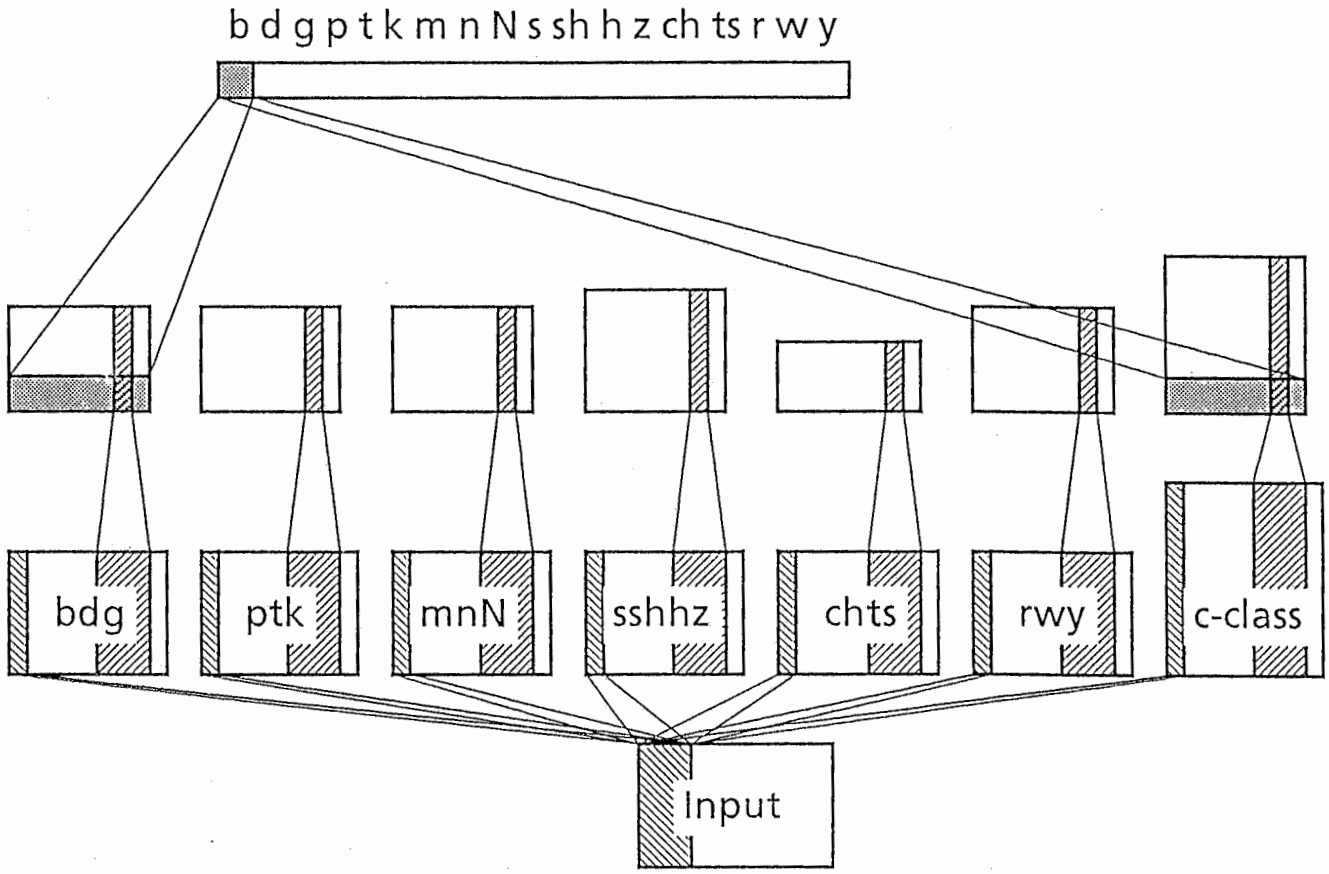

Fig.8. A modular construction of TDNNs for all consonant recognition

5.2.2. All Phoneme Network Architecture

Similarly, our all phoneme TDNN(shown in Fig.9) was modularly constructed from the all consonant network and the vowel network. The outputs with five frame windows from hidden layer 2 were integrated into one frame in hidden

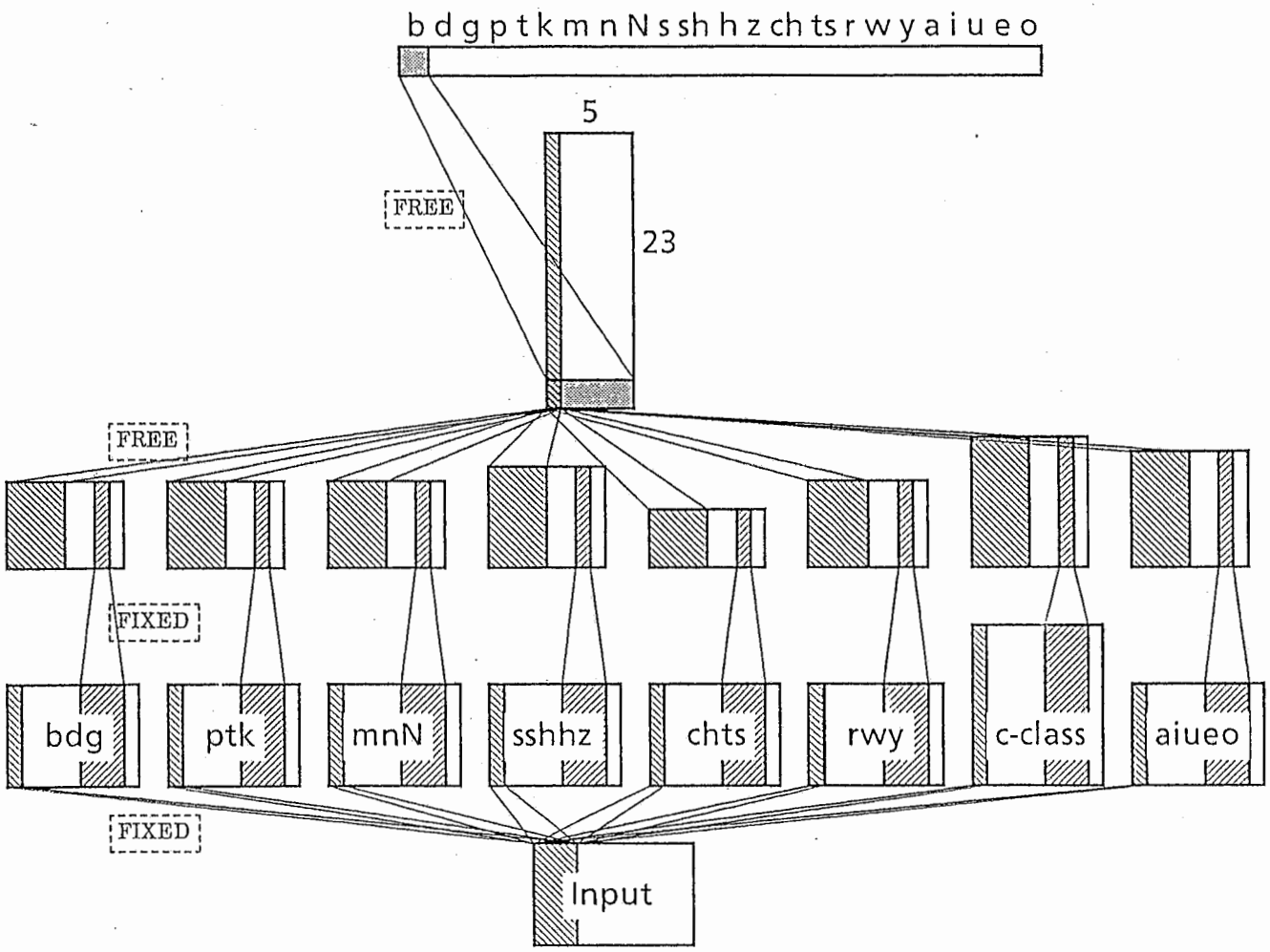

Fig.9. Integrated Modular Construction for an All Phoneme Networks

layer 3, and each unit in hidden layer 3 was integrated into the corresponding output unit[8]. The overall phoneme TDNN has 241 inputs, 23 outputs, 1,628 total units and 79,281 connections. Its size is somewhat smaller than that of the hierarchical structure of TDNNs in Fig.6. The training was done with small random weights between hidden layers 2 and 3, and the output layer, and with frozen weights from the input layer to hidden layer 2. Similar to the training of

the all consonant network, after training all free weights, all weights were free and fine-tuned. Training took only about 1.5 hours with our fast back-propagation learning methods using 200 tokens per phoneme category (total tokens = 4,600).

5.2.3. Results of Phoneme Recognition

After completion of the learning run the consonant net was evaluated over

Table 3 : Phoneme Recognition Performance Results.

Task	Recognition Rate (%)
All consonant TDNN	95.0
All-Net Fine Tuning	96.0
HMM (standard)	83.6
HMM (improved)	93.8
All phoneme TDNN	94.7

3,061 consonant test tokens, and achieveda 95.0% recognition accuracy. All-net fine tuning was then performed by freeing up *all* connections in the network to allow all connections to make small additional adjustments in the interest of better overall performance. After completion of all-net fine tuning, the performance of the network then yielded 96.0% correct consonant recognition over the test data. Table 3 summarizes the results for the consonant and phoneme recognition tasks. Our all phoneme net yielded a phoneme recognition score of 94.7% over the test data. To put these recognition results into perspective, we have also compared these results with several implementations of a Hidden Markov Model trained to perform the same consonant task. Two entries are shown in Table 3. The first (83.6%) shows the consonant recognition performance of a relatively standard (although optimized [12,13,3]) HMM. Recently, a set of additional techniques (shown here as "improved HMM") yielded substantial gains in performance [14]. They include use of three separate codebooks based on Weighted Likelihood Ratio (WLR), Differential Cepstral Coefficients and Power[14] in order to better represent the dynamic properties of speech events (such as transitions, bursts, etc.). Substantial differences therefore exist between the input representations used by the two methods. However, as they were both developed in good faith by two separate research groups attempting to optimize

their respective model, we believe they still provide an insightful comparison. Our results indicate that the TDNN yields significantly lower error rates.

5.3. Results of Spotting Phonemes Using the Large TDNN

Spotting results are shown in Table 4 together with the hierarchical spotting results. Similar hierarchical spotting results such as a correct spotting rate of 90.8%, an omission error rate of 9.2% and an insertion error rate of 102.0% were obtained. The omission error was somewhat more, and the insertion error was somewhat less than those of the hierarchical technique, because the hierarchical approach isn't likely to initiate the lateral inhibition between phonemes. The length of phoneme outputs was also shorter than that of the hierarchical phoneme spotting results. We should incrementally train the whole network as well as the hierarchical network using additional tokens in the phoneme boundaries and in the silence interval.

6. Conclusion

We have reported several techniques to resolve obstacles to realizing continuous speech recognition using TDNNs. We have first shown that learning speed for the back-propagation procedure can, to a large extent, be reduced thanks to improvements on *the modeling of the Error surface, learning strategy and control of the weight modifications.* We applied these improved methods to recognizing all phonemes, and achieved *excellent recognition performance* with a small amount of CPU time. We obtained a recognition rate of 96.0% for the all consonant task, and a rate of 94.7% for the all phoneme task. We believe that these good performance results are due to the key properties of TDNNs, including: *shift invariance*, the proper representation of the *dynamic time-varying properties* of speech and the automatic discovery of *alternate, complementary internal features* of speech.

The serious problems associated with scaling smaller phonemic subcomponent networks to larger phonemic tasks are overcome by careful modular design. Modular design is achieved by several important strategies: *selective and*

Table 4: Phoneme spotting results

	Hierarchical net		Whole net
	phon.goup	phoneme	phoneme
correct rate	330/347 (95.1%)	319/347 (91.9%)	315/347 (90.8%)
omission error	17/347 (4.9%)	28/347 (8.1%)	32/347 (9.2%)
insertion error	279/347 (80.4%)	438/347 (126.2%)	354/347 (102.0%)

incremental learning of subcomponent tasks, *exploitation of previously learned hidden structure*, the application of *connectionist glue* or class distinctive features to allow for separate networks to "grow" together, *partial training* of portions of a larger net and finally, *all-net fine tuning* for making small additional adjustments in a large net.

We have performed two kinds of spotting experiments: CV-syllable spotting and phoneme spotting, and showed that high performance Japanese CV-syllable and phoneme spotting are indeed possible using the TDNNs. The syllable spotting approach uses a simple TDNN architecture and is easily extended to other CV-syllable networks only by making training tokens of a specific syllable and its easily confused syllables.

The spoting experiments on "BA" syllables showed that a rate of 96.7% was achieved, and other possible syllables except "BA" (not only "DA", "GA", "PA", "TA" and "KA" but also all other syllables) are well inhibited at a rate of 99.3%.

On the other hand, in phoneme spotting, a hierarchical TDNN architecture was applied to input word utterances where phoneme group spotting was then performed as a first step, and phoneme discrimination was performed within the determined phoneme group. The excellent spotting performance of 92% correct was obtained.

For comparison with the hierarchical approach, we spotted all phonemes at the same time using the large phonemic TDNN scaled from smaller subnetworks. The spotting performance was identical to that of the hierarchical structure of TDNN.

These results in spotting Japanese-CV syllables and phonemes in words strongly suggested that these spotting techniques can be applied to recognizing not only spoken words but also continuous speech.

Acknowledgement

The authors would like to express their gratitude to Dr.Akira Kurematsu, president of ATR Interpreting Telephony Research Laboratories, for his encouragement and support, and to Ms. Kitaoka who helped prepare the graphic tools for displaying the results of the spotting experiments. We are also indebted to the members of the Speech Processing Department at ATR, for their constant help in the various stages of this research.

References

[1] R.E.Rumelhart and J.L.McClelland, "Parallel Distributed Processing; Explorations in the Microstructure of Cognition", Volume I and II, MIT Press, Cambridge, MA, 1986.

[2] A.Waibel, T.Hanazawa, G.Hinton, K.Shikano, and K.Lang, "Phoneme Recognition: Neural Networks vs. Hidden Markov Models", International Conference on Acoustics, Speech, and Signal Processing, Apr. 1988.

[3] A Waibel, T. Hanazawa, G. Hinton, K.Shikano, and Lang K. Phoneme recognition using time-delay neural networks. IEEE, Transactions on Acoustics, Speech and Signal Processing, 1988. (in press).

[4] C.Kamm, T.Landauer and S. Singhal, "Using an Adaptive Network to Recognize Demisyllables in Continuous Speech", J.Acoust. Soc. Amer., vol.83, suppl. 1. X7, May 1988.

[5] R.P.Lipmann, "An Introduction to Computing with Neural Nets", IEEE ASSP Magazine, 4-22, Apr. 1987.

[6] P.Haffner, A.Waibel, H.Sawai and K.Shikano, "Fast Back-Propagation Learning Methods for Neural Networks in Speech", Technical Report TR-I-0058, ATR Interpreting Telephony Research Laboratories, Nov. 1988.

[7] A.Waibel, H.Sawai, K.Shikano, "Modularity and Scaling in Large Phonemic Neural Networks", Technical Report TR-I-0034, ATR Interpreting Telephony Research Laboratories, Aug. 1988.

[8] H.Sawai, A.Waibel, M.Miyatake and K.Shikano, "Phoneme Recognition by Scaling up Modular Time-Delay Neural Networks", In IEICE techinical report, Dec.1988.

[9] R.E.Rumelhart, "Learning and Generalization: The role of Minimal Networks", in Proceedings of the ATR Workshop on Neural Networks and PDP, Osaka, July 1988.

[10] Y. Sagisaka, K. Takeda, S. Katagiri, and H. Kuwabara, "Japanese Speech Dataabase with Fine Acoustic-Phonetic Transcriptions", Technical Report, ATR Interpreting Telephony Research Laboratories, May 1987.

[11] S.E. Fahlman, *"An Empirical Sudy of Learning Speed in Back-Propagation Networks"*, Technical Report CMU-CS-88-162, Carnegie-Mellon University, June 1988.

[12] T. Hanazawa, T. Kawabata, and K. Shikano, "Discrimination of Japanese voiced stops using Hidden Markov Model", In *Conference of the Acoustical Society of Japan*, pages 19-20, October 1987. (in Japanese).

[13] T. Hanazawa, T. Kawabata, and K. Shikano, "Recognition of Japanese voiced stops using Hidden Markov Models", In *IEICE technical report*, December 1987. (in Japanese).

[14] T. Hanazawa. Personal communication. unpublished, 1988.

[15] R.L.Watrous, "Learning Algorithms for Connectionist Networks: Applied Gradient Methods for Non-Linear Optimization", In Proceedings of the IEEE International Conference on Neural Networks, pp619-627, San Diego, CA, 1987.