TR - I - 0034

Modularity and Scaling in Large Phonemic Neural Networks

Alex Waibel, Hidefumi Sawai, Kiyohiro Shikano

ATR Interpreting Telephony Research Laboratories

August 5, 1988

Contents

1 **Introduction** 6

2 **Small Phonemic Classes by Time-Delay Neural Networks** 6
- 2.1 Review of a Time-Delay Neural Network's Architecture 7
- 2.2 Discrimination Performance in Phonemic Subclasses 11

3 **Scaling TDNNs to Larger Phonemic Classes** 11
- 3.1 The Problem of Training Time 11
- 3.2 Experiments with Modularity 14
- 3.3 Steps for the Design of Large Scale Neural Nets 16

4 **Consonant Recognition by Modular TDNN Design** 20
- 4.1 Consonant Network Architecture 20
- 4.2 Results 20

5 **Conclusion** 23

List of Figures

1 A Time Delay Neural Network (TDNN) unit 8
2 The TDNN architecture (input: "BA") 9
3 TDNN activation patterns for a BDGPTK-net 13
4 BDGPTK-net trained from hidden units from a BDG- and a PTK-net. 15
5 Combination of a BDG-net, a PTK-net and a class distinctive Voiced/Unvoiced-net. . . . 17
6 Combination of a BDG-net and a PTK-net using 4 additional units in hidden layer 1 as free connectionist "glue". 18
7 Modular Construction of All Consonant Network 21

List of Tables

1 Recognition results for 7 phoneme classes 12
2 From BDG to BDGPTK; Modular Scaling Methods. 19
3 Consonant Recognition Performance Results. 22

Abstract

Scaling connectionist models to larger connectionist systems is difficult, because larger networks require increasing amounts of training time and data and the complexity of the optimization task quickly reaches computationally unmanageable proportions. In this paper, we train several small Time-Delay Neural Networks aimed at all phonemic subcategories (nasals, fricatives, etc.) and report excellent fine phonemic discrimination performance for all cases. Exploiting the hidden structure of these smaller phonemic subcategory networks, we then propose several techniques that allow us to "grow" larger nets in an incremental and modular fashion without loss in recognition performance and without the need for excessive training time or additional data. These techniques include *class discriminatory learning, connectionist glue, selective/partial learning and all-net fine tuning.* A set of experiments shows that stop consonant networks (BDGPTK) constructed from subcomponent BDG- and PTK-nets achieved up to 98.6% correct recognition compared to 98.3% and 98.7% correct for the component BDG- and PTK-nets. Similarly, an incrementally trained network aimed at *all* consonants achieved recognition scores of about 95.9% correct. These result were found to be comparable to the performance of the subcomponent networks and significantly better than several alternative speech recognition methods.

Acknowledgement

The authors would like to express their gratitude to Dr. Akira Kurematsu, president of ATR Interpreting Telephony Research Laboratories, for his enthusiastic encouragement and support, which made this research possible. We are also indebted to the members of the Speech Processing Department at ATR, for their help in the various stages of this research.

1 Introduction

A number of studies have recently demonstrated [1,2,3] that connectionist architectures capable of capturing some critical aspects of the dynamic nature of speech, can achieve superior recognition performance for small but difficult phonemic discrimination tasks. Encouraged by these results we would like to explore the question, how we might expand on these models to make them useful for the design of speech recognition systems. A problem that emerges, however, as we attempt to apply neural network models to the full speech recognition problem is the problem of scaling. Simply extending our networks to ever larger structures and retraining them soon exceeds the capabilities of even the fastest and largest of today's supercomputers. Moreover, the search complexity of finding an optimal solution in a huge space of possible network configurations quickly assumes unmanageable proportions. In an effort to extend our models from small recognition tasks to large scale speech recognition systems, we must therefore explore modularity and incremental learning as design strategies to break up a large learning task into smaller subtasks. Breaking up large tasks into subtasks to be tackled by individual black boxes interconnected in ad hoc arrangements, on the other hand, would mean to abandon one of the most attractive aspects of connectionism: the ability to perform complex constraint satisfaction tasks in a massively parallel and interconnected fashion, in view of an overall optimal performance goal. In this paper we demonstrate based on a set of experiments aimed at phoneme recognition that it is indeed possible to construct large neural networks by exploiting the hidden structure of smaller trained subcomponent networks. A set of successful techniques is developed that bring the design of practical large scale connectionist recognition systems within the reach of today's technology.

The present paper has five parts: In the next section we review Time-Delay Neural Networks as a technique to achieve accurate, reliable classification of phonemes in small but ambiguous phonemic subcategories (e.g., BDG, PTK, etc.). Excellent performance results are reported for *all* phonemic coarse classes found in a Japanese large vocabulary word database. In section 3, we then explore techniques for the modular extension of small networks to larger "connectionist systems". In section 4, we then apply these techniques and present a large network, that was designed to recognize *all* the consonants in our database. We summarize our results in the last section of this paper.

2 Small Phonemic Classes by Time-Delay Neural Networks

To be useful for the proper classification of speech signals, a neural network must have a number of properties. First, it should have multiple layers and sufficient interconnections between units in each of these layers. This is to ensure that the network will have the ability to learn complex non-linear decision surfaces[4]. Second, the network should have the ability to represent relationships between events in time. These events could be spectral coefficients, but might also be the output of higher level feature detectors. Third, the actual features or abstractions learned by the network should be invariant under translation in time. Fourth, the learning procedure should not require precise temporal alignment of the labels that are to be learned. Fifth, the number of weights in the network should be small compared to the amount of training data so that the network is forced to encode the training data by extracting regularity. In the following, we review Time-Delay Neural Networks (TDNNs) as an architecture that satisfies all of these criteria and was designed explicitly for the classification of phonemes within small phonemic classes such as the voiced stops, "B", "D", "G", the voiceless stops "P", "T", "K", etc.

2.1 Review of a Time-Delay Neural Network's Architecture

The basic unit used in many neural networks computes the weighted sum of its inputs and then passes this sum through a non-linear function, most commonly a threshold or sigmoid function[4,5]. In our TDNN, this basic unit is modified by introducing delays D_1 through D_N as shown in Fig.1. The J inputs of such a unit now will be multiplied by several weights, one for each delay and one for the undelayed input. For N = 2, and J = 16, for example, 48 weights will be needed to compute the weighted sum of the 16 inputs, with each input now measured at three different points in time. In this way a TDNN unit has the ability to relate and compare current input with the past history of events. The sigmoid function was chosen as the non-linear output function F due to its convenient mathematical properties[5,6].

For the recognition of phonemes, a three layer net is constructed. Its overall architecture and a typical set of activities in the units are shown in Fig.2 based on one of the phonemic subcategory tasks (BDG).

At the lowest level, 16 melscale spectral coefficients serve as input 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[1,2] and adjacent coefficients in time collapsed resulting in an overall 10 msec frame rate. The coefficients of an input token (in this case 15 frames of speech centered around the hand labeled vowel onset) were then normalized to lie between -1.0 and +1.0 with the average at 0.0. Fig.2 shows the resulting coefficients for the speech token "BA" as input to the network, where positive values are shown as black and negative values as grey squares.

This input layer is then fully interconnected to a layer of 8 time delay hidden units, where J = 16 and N = 2 (i.e., 16 coefficients over three frames with time delay 0, 1 and 2). An alternative way of seeing this is depicted in Fig.2. It shows the inputs to these time delay units expanded out spatially into a 3 frame window, which is passed over the input spectrogram. Each unit in the first hidden layer now receives input (via 48 weighted connections) from the coefficients in the 3 frame window. The particular delay choices were motivated by earlier studies[7,1,2,8,9,10,11].

In the second hidden layer, each of 3 TDNN units looks at a 5 frame window of activity levels in hidden layer 1 (i.e., J = 8, N = 4). The choice of a larger 5 frame window in this layer was motivated by the intuition that higher level units should learn to make decisions over a wider range in time based on more local abstractions at lower levels.

Finally, the output is obtained by integrating (summing) the evidence from each of the 3 units in hidden layer 2 over time and connecting it to its pertinent output unit (shown in Fig.2 over 9 frames for the "B" output unit). In practice, this summation is implemented simply as another TDNN unit which has fixed equal weights to a row of unit firings over time in hidden layer 2. While the network shown in Fig.2 was designed for a 3 class problem (e.g., BDG or PTK), variations to accommodate 2, 4 or 5 classes are easily implemented by allowing for 2, 4 or 5 units in hidden layer 2 and in the output layer.

When the TDNN has learned its internal representation, it performs recognition by passing input speech over the TDNN units. In terms of the illustration of Fig.2 this is equivalent to passing the time delay windows over the lower level units' firing patterns. At the lowest level, these firing patterns simply consist of the sensory input, i.e., the spectral coefficients.

Each TDNN unit outlined in this section has the ability to encode temporal relationships within the range of the N delays. Higher layers can attend to larger time spans, so local short duration features will be formed at the lower layer and more complex longer duration features at the higher layer. The learning procedure ensures that each of the units in each layer has its weights adjusted in a way that improves the network's overall performance.

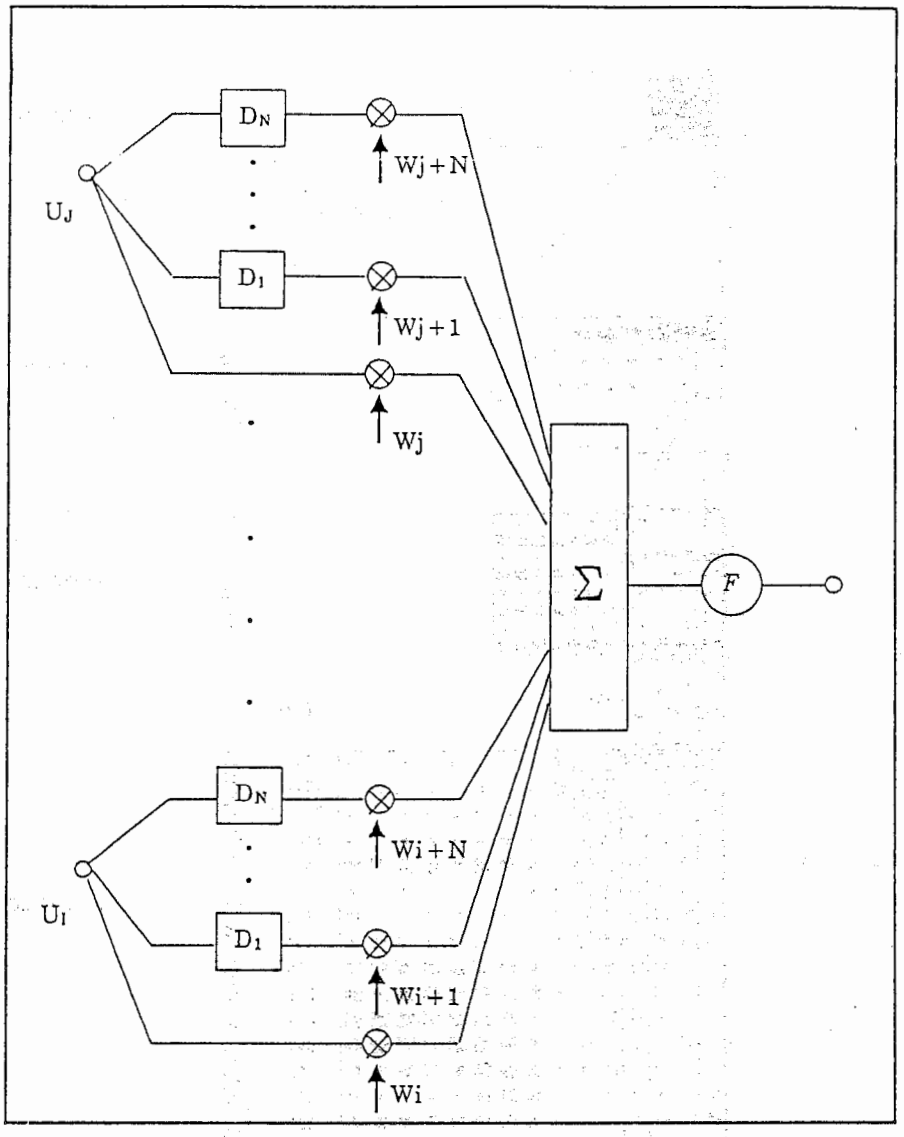

Figure 1: A Time Delay Neural Network (TDNN) unit

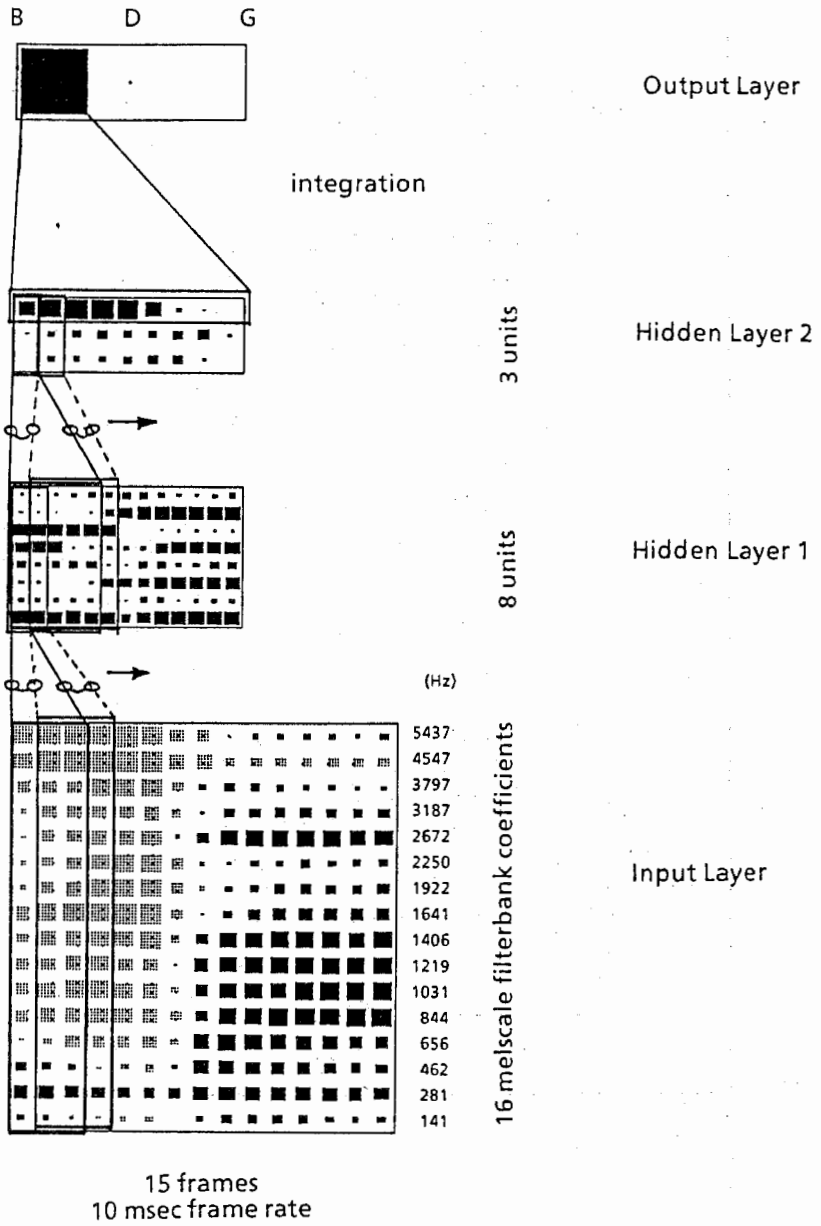

Figure 2: The TDNN architecture (input: "BA")

The network described is trained using the Back-propagation Learning Procedure[5,6]. This procedure iteratively adjusts all the weights in the network so as to decrease the error obtained at its output units. For translation invariance, we need to ensure during learning that the network is exposed to *sequences* of patterns and that it is allowed (or encouraged) to learn about the most powerful cues and sequences of cues among them. Conceptually, the back-propagation procedure is applied to speech patterns that are stepped through in time. An equivalent way of achieving this result is to use a spatially expanded input pattern, i.e., a spectrogram plus some constraints on the weights. Each collection of TDNN-units described above is duplicated for each one frame shift in time. In this way the whole history of activities is available at once. Since the shifted copies of the TDNN-units are mere duplicates and are to look for the same acoustic event, the weights of the corresponding connections in the time shifted copies must be constrained to be the same. To realize this, we first apply the regular back-propagation forward and backward pass to all time shifted copies as if they were separate events. This yields different error derivatives for corresponding (time shifted) connections. Rather than changing the weights on time-shifted connections separately, however, we actually update each weight on corresponding connections by the same value, namely by *the average* of all corresponding time-delayed weight changes[1]. Fig.2 illustrates this by showing in each layer only two connections that are linked to (constrained to have the same value as) their time shifted neighbors. Of course, this applies to all connections and all time shifts. In this way, the network is forced to discover useful acoustic-phonetic features in the input, regardless of when in time they actually occurred. This is an important property, as it makes the network independent of errorprone preprocessing algorithms, that otherwise would be needed for time alignment and/or segmentation.

Experimental Conditions, Database For performance evaluation, we have used a large vocabulary database of 5240 common Japanese words[1,2]. The data used in this paper was uttered in isolation by one male native Japanese speaker (MAU). All utterances were recorded in a sound proof booth and digitized at a 12 kHz sampling rate. The database was then split into a training set and a testing set of 2620 utterances each, from which the actual phonetic tokens were extracted. The training tokens (up to 600 tokens per phoneme[2]) were randomized within each phoneme class. For a given training run they were then presented, alternating between each class to be learned. If a phoneme class was represented by an insufficient number of available training tokens, random tokens from its set were repeated, in order to preserve the alternating sequence of presentations among all training tokens. For performance evaluation, we have run all experiments on the testing tokens only, i.e., on tokens *not* included during training.

The entire database was phonetically handlabeled[12]. These labels were used in the experiments reported below to center a given phoneme in the input range used for learning and evaluation. No attempt was made to correct for improper handlabels. Since all networks described here were trained in a translation invariant fashion, possible misalignments at the input are of no serious concern as long as all the critical features needed for discrimination are present *somewhere* in the input range. For consistency among our networks and efficiency of learning, we continued to employ a 150 msec input range. Note, however, that longer input ranges are possible and might in fact be preferable to extract all useful features of a given phoneme. All tokens in the database were included in the test set or the training

[1] Note that weight changes were carried out after presentation of all training samples[6].

[2] Note, that for some phoneme categories an unnecessarily large number of tokens was found in the database (e.g., vowels), while for some others (e.g., "P") only few tokens were extracted. While excessive tokens are simply discarded at random to reduce the dataset size, a lack of tokens leads to poor generalization. The low recognition scores for "P" are therefore a result of the limited training data.

set, respectively, and no preselection was done. The resulting data included a considerable amount of variability (see [1,2] for examples) due to its position within an utterance or phonetic context.

2.2 Discrimination Performance in Phonemic Subclasses

To evaluate our TDNNs on all phoneme classes (see [1,2] for in depth discussion for voiced stops), recognition experiments have been carried out for seven phonemic subclasses found in the database. For each of these classes, TDNNs with an architecture similar to the one shown in Fig.2 were trained. A total of seven nets aimed at the major coarse phonetic classes in Japanese were trained, including voiced stops B, D, G, voiceless stops P,T,K, the nasals M, N and syllabic nasals, fricatives S, SH, H and Z, affricates CH, TS, liquids and glides R, W, Y and finally the set of vowels A, I, U, E and O. Each of these nets was given between two and five phoneme classes to distinguish and the pertinent input data was presented for learning. Note, that each net was trained only within each respective coarse class and has no notion of phonemes from other classes yet. Table 1 shows the recognition results for each of these major coarse classes.

3 Scaling TDNNs to Larger Phonemic Classes

We have seen in the previous section that TDNNs achieve superior recognition performance on difficult but small recognition tasks. To train these networks, however, substantial computational resources were needed. This raises the question of how our good but admittedly limited networks could be extended to encompass *all* phonemes or handle speech recognition in general. To shed light on this question of scaling, we consider first the problem of extending our networks from the task of voiced stop consonant recognition (hence the BDG-task) to the task of distinguishing among *all* stop consonants (the BDGPTK-task).

3.1 The Problem of Training Time

For a network aimed at the discrimination of the voiced stops (a BDG-net), approximately 6000 connections had to be trained over about 800 training tokens. An identical net (also with approximately 6000 connections to be trained[3]) 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. We have trained such a network for experimental purposes. To allow for the necessary number of features to develop we have given this net 20 units in the first hidden layer, 6 units in hidden layer 2 and 6 output units. Fig.3 shows this net in actual operation with a "G" presented at its input. Eventually a high performance network was obtained that achieves 98.3% correct recognition over a 1613-token BDGPTK-test database, but 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 the number of connections has 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

[3]Note, that these are connections over which a back-propagation pass is performed during each iteration. Since many of them share the same weights, only a small fraction (about 500) of them are actually free parameters.

phoneme	TDNN		
	#errors/ #tokens	%correct	total %
b	5/227	97.8	98.6
d	2/179	98.9	
g	2/252	99.2	
p	6/15	60.0	98.7
t	6/440	98.6	
k	0/500	100.0	
m	14/481	97.1	96.6
n	16/265	94.0	
N	12/488	97.5	
s	6/538	98.9	99.3
sh	0/316	100.0	
h	1/207	99.5	
z	1/115	99.1	
ch	0/123	100.0	100
ts	0/177	100.0	
r	0/722	100.0	99.9
w	0/78	100.0	
y	1/174	99.4	
a	0/600	100.0	98.6
i	1/600	99.8	
u	25/600	95.8	
e	8/600	98.7	
o	7/600	98.8	

Table 1: Recognition Results for 7 Phoneme Classes

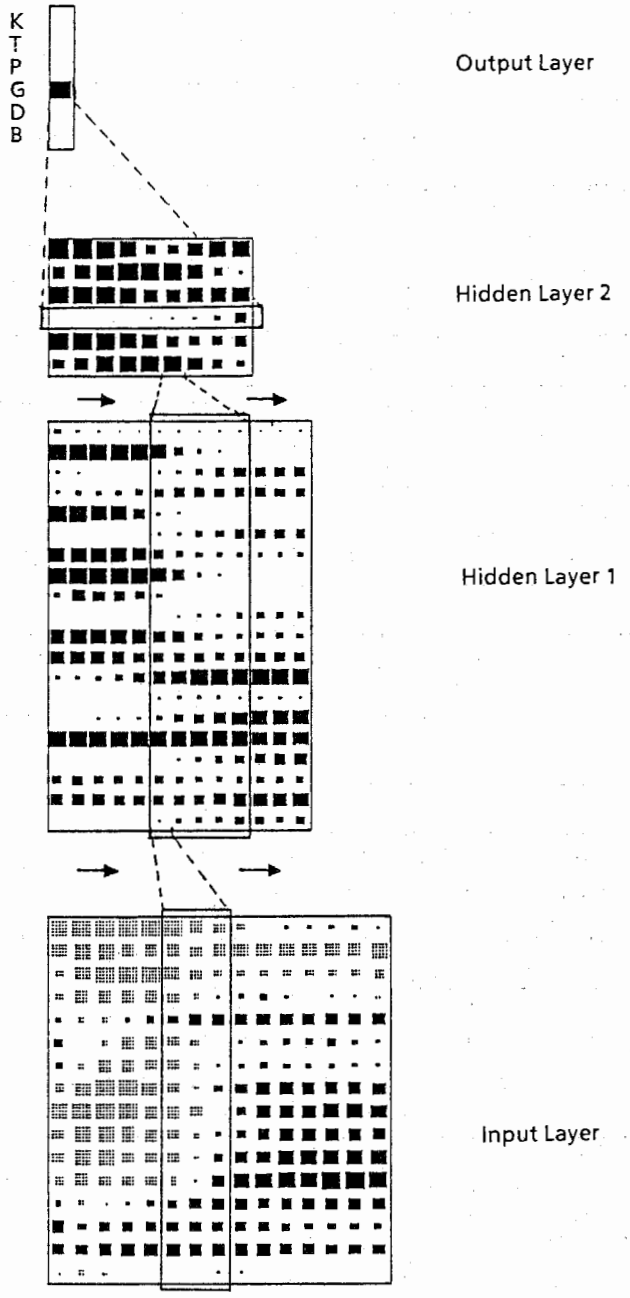

Figure 3: TDNN activation patterns for a BDGPTK-net

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. Despite progress towards faster learning algorithms[13,14], it is clear that we cannot hope for one single monolithic network to be trained within reasonable time as we increase task size and eventually aim for continuous, speaker-independent speech recognition. Moreover, requiring that all classes be considered and samples of each class be presented during training, is undesirable for practical reasons as we contemplate the design of large neural systems. Alternative ways to modularly construct and incremental train such large neural systems must therefore be explored.

3.2 Experiments with Modularity

Four experiments were performed to explore methodologies for constructing phonetic neural nets from smaller component subnets. As a task we used again stop consonant recognition although other tasks have recently been explored with similar success (BDG and MNsN). As in the previous section we used a large database of 5240 common Japanese words spoken in isolation. Half of these utterances were used as training database, and the other half for testing. The two component phoneme classes that make up the set of stops are given by the voiced stops B,D and G (the BDG-set) and the voiceless stops P,T and K (the PTK-set).

A First Attempt Two separate TDNNs have been trained for the two sets based on training data from their own set only. On testing data the BDG-net used here performed 98.3% correct for the BDG-set and the PTK-net achieved 98.7% correct recognition for the PTK-set [4]. As a first naive attempt we have now simply run a speech token from either set (i.e., B,D,G,P,T or K) through both a BDG-net and a PTK-net and selected the class with the *highest activation* from either net as the recognition result. As might have been expected (the component nets had only been trained for their respective classes), poor recognition performance (60.5%) resulted from the 6 class experiment. This is partially due to the inhibitory property of the TDNN that we have observed elsewhere[1,2]. While this property results in high confidence and reliability at the output decisions for a token from a class the network was trained for, it also produces erroneous activations for classes that had not been part of its world. To combine the two networks more effectively, therefore, portions of the net have to be retrained.

Exploiting the Hidden Structure of Subcomponent Nets We start by assuming that the first hidden layer in either net already contains all the lower level acoustic phonetic features we need for proper identification of the stops and freeze the connections from the input layer (the speech data) to the first hidden layer's 8 units in the BDG-net and the 8 units in the PTK-net. Back-propagation learning is then performed only on the connections between these 16 (= 2 X 8) units in hidden layer 1 and hidden layer 2 and between hidden layer 2 and the combined BDGPTK-net's output. This network is shown in Fig.4 with a "G" token presented as input. Only about 4,400 new connections had to be trained in this case and the resulting network achieved a recognition performance of 98.1% over the testing data. Combination of the two subnets has therefore yielded a promising combined net although a slight performance degradation compared to the subnets was observed. This degradation could be explained by the increased complexity

[4] The connection weights used in these experiments stem from a shorter learning run than the one reported in the previous section and elsewhere[1], hence the slightly different recognition scores.

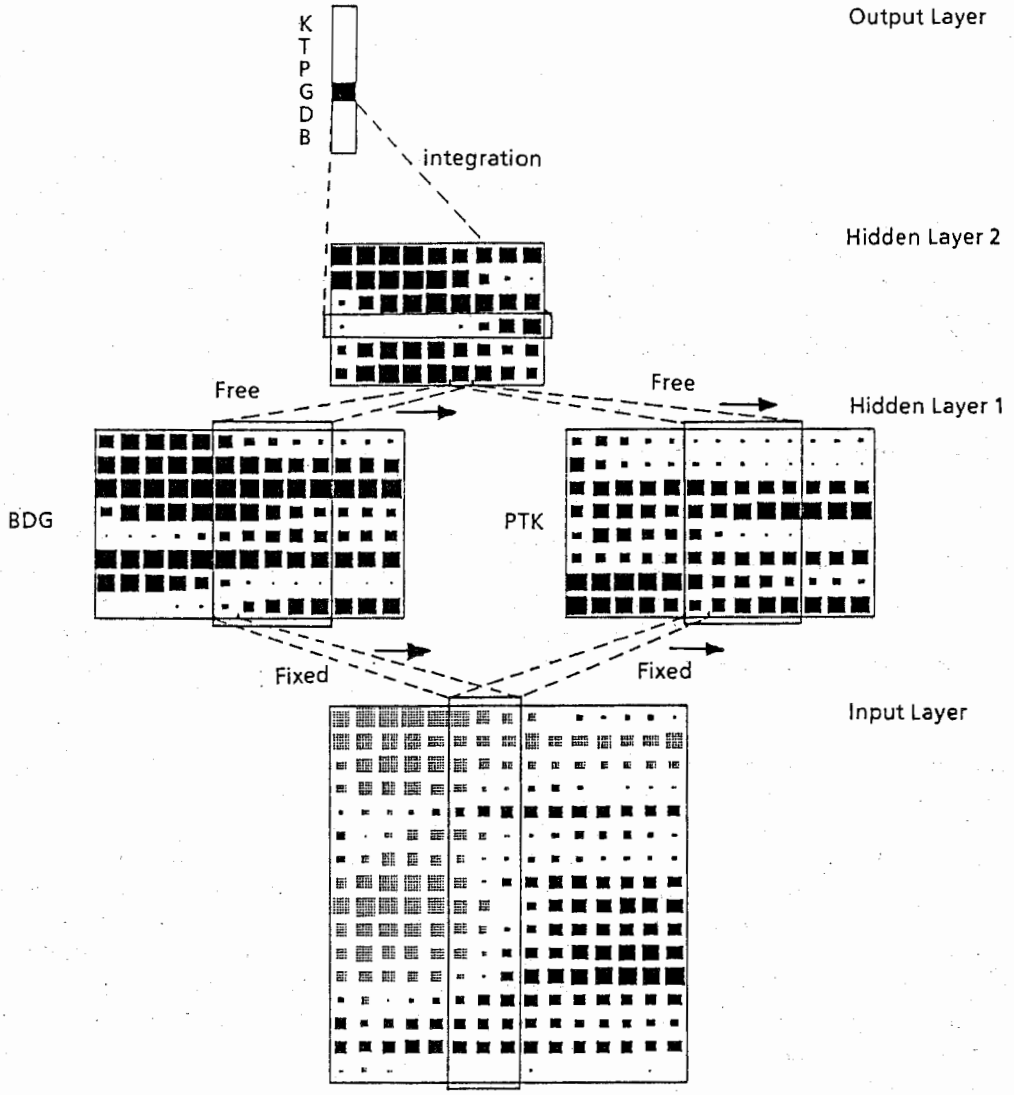

Figure 4: BDGPTK-net trained from hidden units from a BDG- and a PTK-net.

of the task, but also by the inability of this net to develop lower level acoustic-phonetic features in hidden layer 1. Such features may in fact be needed for discrimination *between* the two stop classes, in addition to the within-class features.

Class Distinctive Features In a third experiment, we therefore first train a separate TDNN to perform the voiced/unvoiced (V/UV) distinction between the BDG- and the PTK-task. The network has a very similar structure as our BDG-nets, except that only four hidden units were used in hidden layer 1 and two in hidden layer 2 and at the output. This V/UV-net achieved better than 99% voiced/unvoiced classification on the test data and its hidden units developed in the process are now used as additional features for the BDGPTK-task. Fig.5 shows the resulting network. As can be seen the connections from the input to the first hidden layer of the BDG-, the PTK- and the V/UV nets are frozen and only the connections that combine the 20 units in hidden layer 1 to the higher layers are retrained. The resulting net was evaluated as before on our testing database and achieved a recognition score of 98.4% correct.

Incremental Learning by Way of "Connectionist Glue" In the previous experiment, good results could be obtained by adding units that we *believed* to be the useful class distinctive features that were missing in our second experiment. In a fourth experiment, we have now examined an approach that allows for the network to be free to discover *any* additional features that might be useful to merge the two component networks. In stead of previously training a class distinctive network, we now add four units to hidden layer 1, whose connections to the input are free to learn any missing discriminatory features to supplement the 16 frozen BDG and PTK features. We call these units the "*connectionist glue*" that we apply to merge two distinct networks into a new combined net. This network is shown in Fig.6. The hidden units of hidden layer 1 from the BDG-net are shown on the left and those from the PTK-net on the right. The connections from the moving input window to these units have been trained individually on BDG- and PTK-data, respectively and -as before- remain fixed during combination learning. In the middle on hidden layer 1 we show the 4 free "Glue" units. Combination learning now finds an optimal combination of the existing BDG- and PTK-features and also supplements these by learning additional interclass discriminatory features. In doing so we have raised the number of connections to be trained to 8,000, which is only a small increase in number of connections (and learning time) over the original component nets. Performance evaluation of this network over the BDGPTK test database yielded a recognition rate of 98.4%.

All-Net Tuning In addition to the techniques described so far, it may be useful to free *all* connections in a large modularly constructed network for an additional small amount of fine tuning. This has been done for the BDGPTK-net shown in Fig.6 yielding some additional performance improvements. The resulting network finally achieved (over testing data) a recognition score of 98.6%.

3.3 Steps for the Design of Large Scale Neural Nets

Table 2 summarizes the major results from our experiments. In the first row it shows the recognition performance of the two initial TDNNs trained individually to perform the BDG- and the PTK-tasks, respectively. Underneath, we show the results from the Hidden Markov Model, as discussed in the previous section. The third row shows that simply adding TDNNs and selecting the unit with the largest

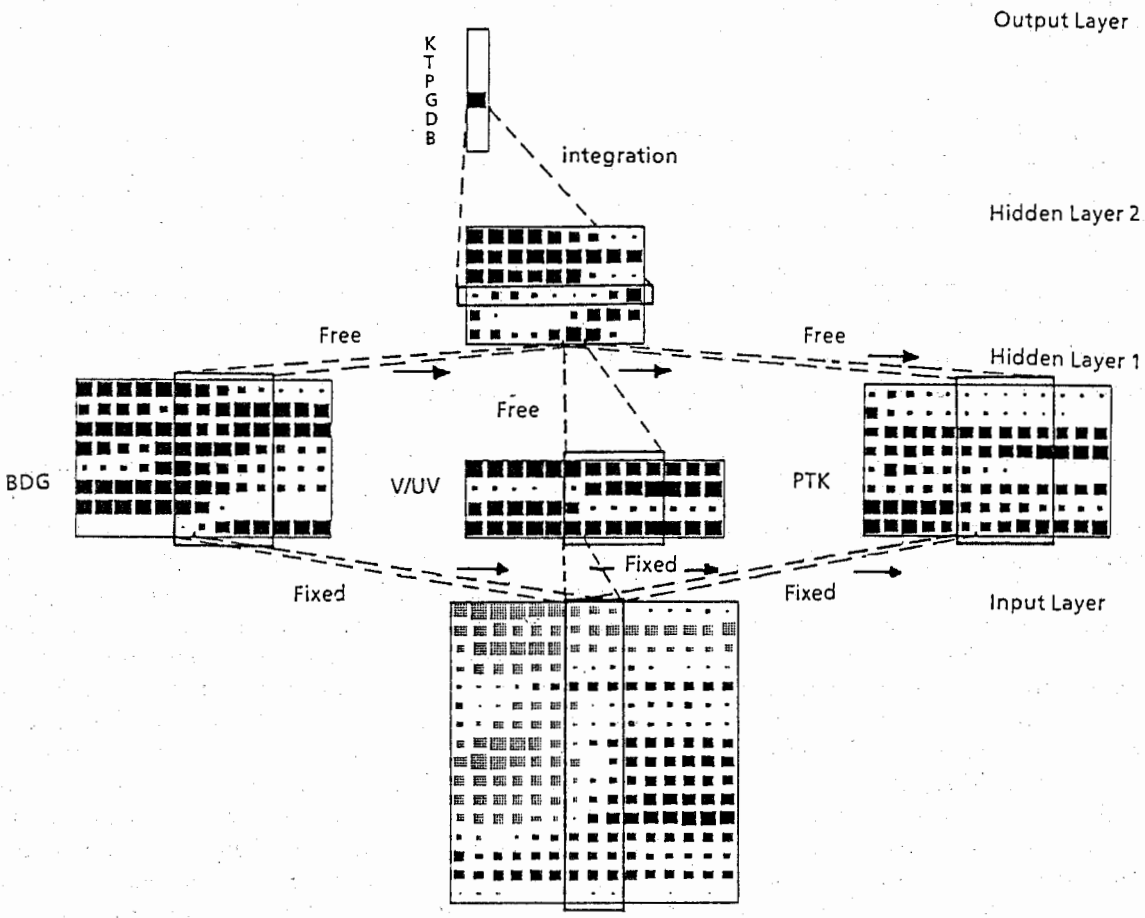

Figure 5: Combination of a BDG-net, a PTK-net and a class distinctive Voiced/Unvoiced-net.

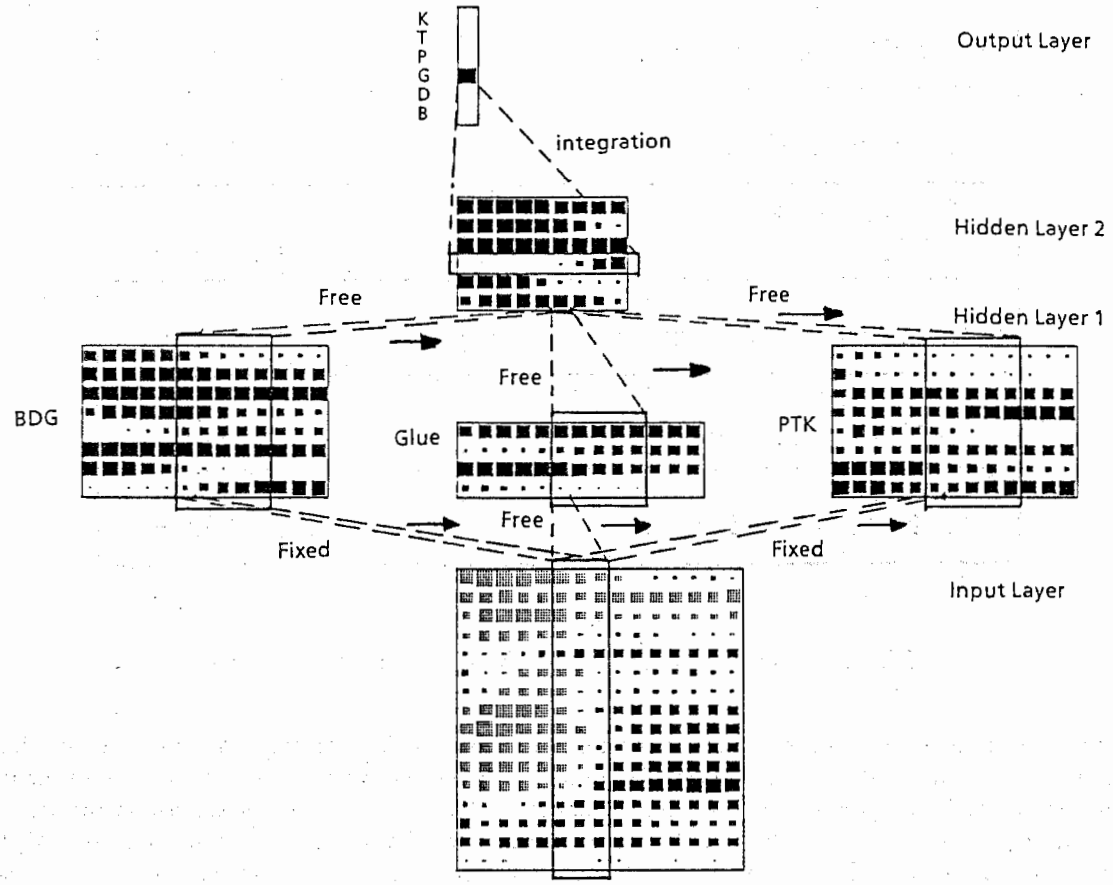

Figure 6: Combination of a BDG-net and a PTK-net using 4 additional units in hidden layer 1 as free connectionist "Glue".

Method	bdg	ptk	bdgptk
Individual TDNNs	98.3 %	98.7 %	
TDNN:Max. Activation			60.5 %
Retrain BDGPTK			98.3 %
Retrain Combined Higher Layers			98.1 %
Retrain with V/UV-units			98.4 %
Retrain with Glue			98.4 %
All-Net Fine Tuning			98.6 %

Table 2: From BDG to BDGPTK; Modular Scaling Methods.

output activation does not lead to acceptable performance (only 60.5% correct). We have observed before that this is in part a negative consequence of inhibition in these networks. While inhibition of incorrect output categories leads to good, robust and confident performance, it creates erroneous results when additional networks are simply added without consideration of the interaction between them. We have then retrained a complete BDGPTK-net which achieves good recognition performance (98.3% correct), but found that it requires excessive amounts of training time. As an alternative, we have then explored three methods that exploit the hidden structure of previously learned subcomponent networks, e.g., the BDG- and PTK-networks. With small additional training at the higher layers these networks could be merged and achieve good recognition performance (98.1%). When additional hidden units from a class distinctive voiced/unvoiced TDNN were added, recognition results improve to 98.4%. Similarly, through the application of "connectionist glue", a 98.4% performance score is achieved. Finally, when all the connections in the latter network are freed to perform small additional adjustments over a few additional training iterations, recognition results improve further to 98.6%.

The results indicate, that larger TDNNs can indeed be trained *incrementally*, without requiring excessive amounts of training and without loss in performance. In fact, the resulting incrementally trained networks appear to perform slightly better than the monolithically trained BDGPTK-net. Moreover, they achieve performance as high as the subcomponent BDG- and PTK-nets alone. As a strategy for the 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.*

4 Consonant Recognition by Modular TDNN Design

The techniques described in the previous section were applied to the task of recognizing *all consonants* in our database. In the following we describe only our first attempts at building such a larger net and note that numerous alternative solutions remain to be explored.

4.1 Consonant Network Architecture

Our consonant TDNN (shown in Fig.7) was constructed modularly from networks aimed at the consonant subcategories described in section 2, 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 was trained that distinguishes *between* the consonant subclasses and thus hopefully provides missing featural information for interclass discrimination much like the V/UV network described in the previous section. 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 hidded 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): one to connect an output unit with the appropriate *interclass* discrimination unit in hidden layer 2, one with the appropriate *intraclass* discrimination unit from hidden layer 2 of the corresponding subcategory net and one with the always activated threshold unit (not shown in Fig.7)[5]. The overall network architecture is illustrated in Fig.7 for the case of an incoming test token (e.g., a "G"). For simplicity, Fig.7 shows only the hidden layers from the BDG-,PTK,SShHZ- and the interclass discrimination nets. At the output, only the two connections leading to the correctly activated "G"-output unit are shown. Units and connections pertaining to the other subcategories as well as connections leading to the 17 other output units are omitted for clarity in Fig.7. All free weights were initialized with small random weights and then trained by the back-propagation learning procedure[6].

4.2 Results

After completion of the learning run the entire net was evaluated over 3061 consonant test tokens, and achieved a 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 95.9% correct consonant recognition over the test data. Table 3 summarizes the our results for the consonant recognition task. In the first 6 rows the recognition results (measured over the available test data in their respective sublasses) are given. The entry "cons.class" shows the performance of the interclass discrimination net in identifying the coarse phonemic subclass of an unknown token. 96.7%

[5] Note, that as before, the time shifted activations of the units in hidden layer 2 are simply integrated and do not receive a separate weight. This was done in the interest of shift-invariance, in order to force the network to learn consonantal features of speech independent of the time alignment implicit in the extraction of the phoneme training tokens.

[6] Since only the top layer was trained in this case, this is equivalent to perceptron learning.

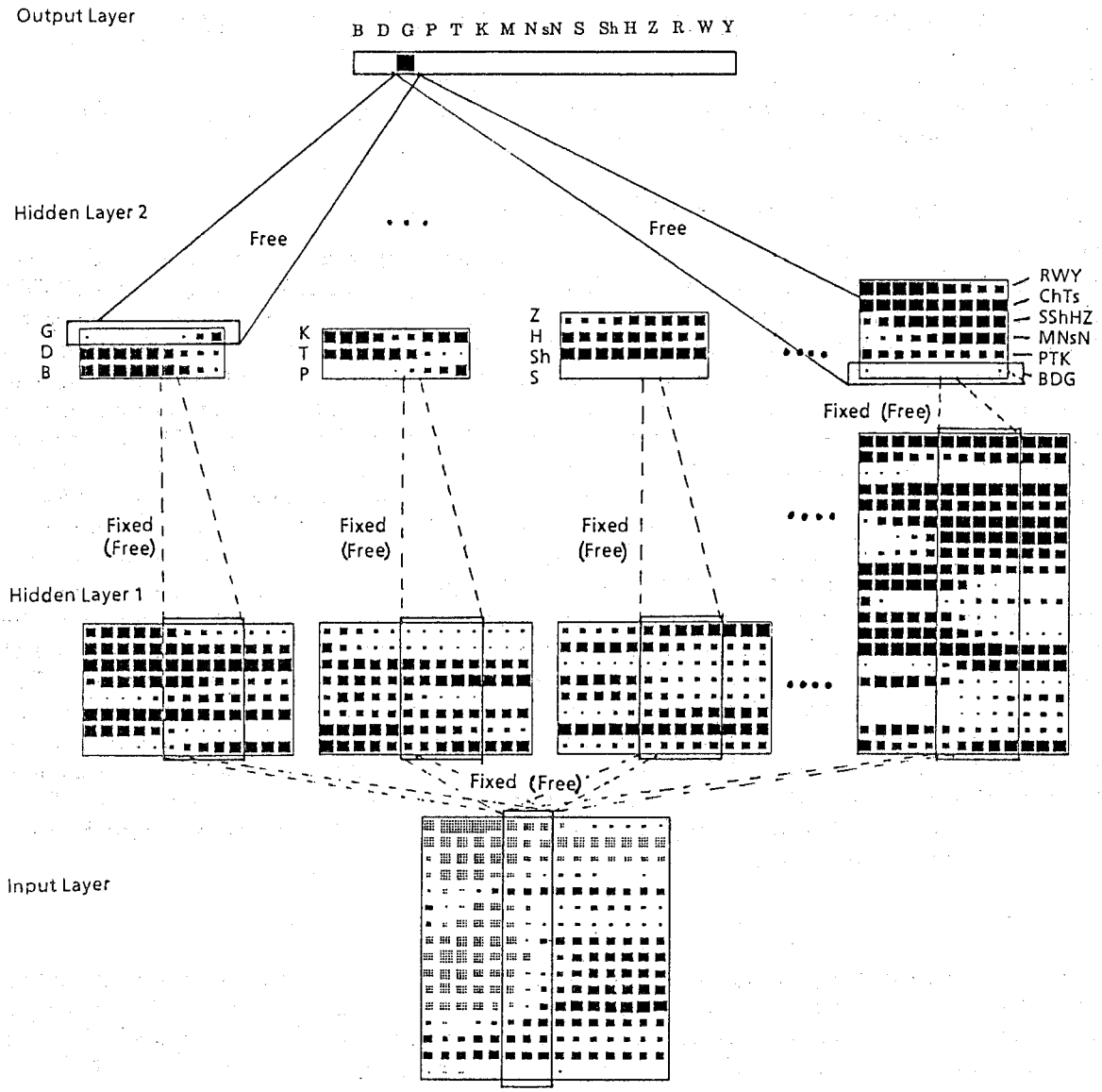

Figure 7: Modular Construction of an All Consonant Network

Consonants

Task	Recognition Rate (%)
bdg	98.6
ptk	98.7
mnN	96.6
sshhz	99.3
chts	100.0
rwy	99.9
cons. class	96.7
All consonant TDNN	95.0
All-Net Fine Tuning	95.9
HMM(standard)	83.6
HMM(improved)	92.7

Table 3: Consonant Recognition Performance Results.

of all tokens were correctly categorized into one of the six consonant subclasses. After combination learning and all-net fine tuning our consonant net then yielded consonant recognition scores of 95.0% and 95.9%, respectively. 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 task. Two entries are shown in table 3. The first (83.6%) shows the recognition performance of a relatively standard (although optimized[15,16,1]) HMM. Recently, a set of additional techniques (shown here as "improved HMM") yielded substantial gains in performance[17]. They include use of three separate codebooks based on Weighted Likelihood Ratio (WLR), Differential Cepstral Coefficients and Power[17] in order to better represent the dynamic properties of speech events (such as transitions, bursts, etc.). In addition, noticeable performance improvements resulted from the use of separate models for phonemes from word initial and word middle positions. These separate models required additional labels in the training data (indicating position within the utterance), that were not given to the TDNNs. 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 either model, we believe they still provide an insightful comparison. Our results indicate that the TDNN yields significantly lower error rates (significant at $p < 0.01$ when compared to the best of our HMMs over an all consonant recognition task.

5 Conclusion

We summarize the major technical results from this work:

- We have reported further experimental results from the use of Time Delay Neural Networks (TDNNs) for recognition in all major phonemic categories in a large vocabulary speech database and have measured *excellent recognition performance*. We believe, that the 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. These properties have been extensively documented elsewhere[1,2].

- 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 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 adjustment in a large net.

- Our techniques have been applied to the construction of a large TDNN aimed at the recognition of *all consonants*. While a number of alternate strategies remain to be explored, our best recognition result so far indicates that the consonants extracted from a large vocabulary database of isolated words can be recognized at a rate of 95.9 % or better using an incrementally trained net. We have compared this performance result with several Hidden Markov Models developed (and improved) in our laboratory and found that the TDNN yielded significantly lower error rates[7]. The results

[7] We would like to caution the reader again, that numerous alternative HMM designs as well as alternative input data representations have not been tried in this comparison. These could lead to further performance improvements. Also while

indicate that a high performing large neural network could indeed be constructed without loss in recognition performance and with only little additional training from smaller networks aimed at smaller subtasks.

Our findings suggest, that judicious application of a number of connectionist design techniques could lead to successful large scale connectionist speech recognition systems.

significantly better recognition performance was achieved by the TDNN for phoneme recognition, word level integration and good word level performance have not been attempted yet. Such integration has been shown successfully using HMMs.

References

[1] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and Lang K. *Phoneme Recognition Using Time-Delay Neural Networks.* Technical Report TR-1-0006, ATR Interpreting Telephony Research Laboratories, October 1987.

[2] 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).

[3] R. Watrous. *Speech Recognition Using Connectionist Networks.* PhD thesis, University of Pennsylvania, September 1988.

[4] R.P. Lippmann. An introduction to computing with neural nets. *IEEE ASSP Magazine*, 4–22, April 1987.

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

[6] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, October 1986.

[7] K. Lang. Connectionist speech recognition. July 1987. PhD thesis proposal, Carnegie-Mellon University.

[8] S. Makino and K. Kido. Phoneme recognition using time spectrum pattern. *Speech Communication*, 225–237, June 1986.

[9] S.E. Blumstein and K.N. Stevens. Acoustic invariance in speech production: evidence from measurements of the spectral characteristics of stop consonants. *Journal of the Acoustical Society of America*, 66:1001–1017, 1979.

[10] S.E. Blumstein and K.N. Stevens. Perceptual invariance and onset spectra for stop consonants in different vowel environments. *Journal of the Acoustical Society of America*, 67:648–662, 1980.

[11] D. Kewley-Port. Time varying features as correlates of place of articulation in stop consonants. *Journal of the Acoustical Society of America*, 73:322–335, 1983.

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

[13] P. Haffner, A. Waibel, and K. Shikano. Fast back-propagation learning methods for neural networks in speech. In *Proceedings of the Fall Meeting of the Acoustical Society of Japan*, October 1988. A more detailed version is in press as ATR-technical report.

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

[15] 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).

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

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