TR－A－0061

63

A Hybrid Speech Recognition System Using HMMs with an LVQ-trained Codebook

LVQ コードブックとHMMを使った
ハイブリッド型音声認識システム

Hitoshi IWAMIDA, Shigeru KATAGIRI, Erik MCDERMOTT and Yoh'ichi TOHKURA

岩見田 均　片桐 滋　マクダモット,E　東倉洋一

1989. 8. 25

ATR視聴覚機構研究所

〒619-02 京都府相楽郡精華町乾谷　☎07749-5-1411

ATR Auditory and Visual Perception Research Laboratories

Inuidani, Seika-cho, Soraku-gun, Kyoto 619-02 Japan

Telephone: +81-7749-5-1411
Facsimile: +81-7749-5-1408
Telex: 5452-516 ATR J

A Hybrid Speech Recognition System Using HMMs with an LVQ-trained Codebook

Hitoshi Iwamida, Shigeru Katagiri, Erik McDermott, and Yoh'ichi Tohkura

ATR Auditory and Visual Perception Research Laboratories
Inuidani, Seika-cho Soraku-gun, Kyoto 619-02, Japan

Abstract

An ongoing area of research concerns the application of connectionist methods to speech recognition. Here, a new speech recognition system using the neurally-inspired Learning Vector Quantization (LVQ) to train HMM codebooks is described. Both LVQ and HMMs are stochastic algorithms holding considerable promise for speech recognition. In particular, LVQ is a vector quantizer with very powerful classification ability, as shown in the high phoneme recognition rates obtained in [McDermott & Katagiri; Proc. of ICASSP 89, 9.S3.1, pp. 81-84, 1989]. HMMs, on the other hand, have the advantage that phone models can easily be concatenated to produce long utterance models, such as word or sentence models. The new algorithm described here combines the advantages inherent in each of these two algorithms. Instead of using a conventional, k-means generated codebook in the HMMs, the new system uses LVQ to adapt the codebook reference vectors so as to minimize the number of errors these reference vectors make when used for nearest neighbor classification of training vectors. The LVQ codebook can then provide the HMMs with high classification power at the phonemic level.

Using a large vocabulary database of 5240 common Japanese words uttered in isolation by a male speaker, two main comparisons were performed to evaluate the LVQ-HMM hybrid:

(1) Comparison of the hybrid algorithm with TDNN [Waibel et. al.; Proc. of ICASSP 88, S3.3, pp.107-110, 1988] and Shift-Tolerant LVQ [McDermott & Katagiri; Proc. of ICASSP 89, 9.S3.1, pp. 81-84, 1989]. Phoneme recognition experiments were performed using the same data as used in TDNN and Shift-Tolerant LVQ, for 7 Japanese phonemic classes. For these tasks, the new algorithm yielded very high performance, comparable to that of TDNN and Shift-Tolerant LVQ.

(2) Comparison of the LVQ-HMM hybrid with HMMs using conventional k-means generated codebooks. Phoneme recognition experiments were performed using phoneme tokens for 25 phoneme categories. In these experiments, the LVQ-HMM hybrid achieved recognition error rates 2 or 3 times lower than those of HMMs using codebooks designed by k-means clustering.

These results demonstrate that by using LVQ instead of k-means to generate HMM codebooks, the high discriminant ability of LVQ can be integrated into an HMM architecture easily extendible to longer utterance models, such as word or sentence models.

1. Introduction

Recently, pattern classification methods using multi-layer perceptrons (for example, TDNN) or the neurally-inspired Learning Vector Quantization (LVQ) have attained excellent performance on phoneme recognition tasks [3, 5]. There have been a number of recent proposals, and some encouraging results, as to how one might extend these highly discriminant models to longer utterance models [15, 16, 17, 18]. This is an ongoing area of research.

Hidden Markov models (HMMs), on the other hand, have been successfully applied to many speech recognition tasks, ranging from phoneme recognition tasks to large vocabulary speaker-independent continuous speech recognition systems [9]. An advantage of HMMs is that it is very easy to extend phone models to long utterance models such as word or even sentence models. On the other hand, HMMs are somewhat lacking in discriminant power, especially when compared to neural network models [5].

There have been a number of recent studies concerning the links between neural nets and HMMs, and suggestions for improving the discriminant ability of HMMs [8, 11, 13, 14]. In the case of LVQ, a very straightforward possibility for integration into an HMM framework easily extendible to word or sentence models is simply to use LVQ to generate a discriminant HMM codebook, instead of the conventional LBG or k-means clustering algorithms. By integrating HMMs and LVQ in this way, we hope to combine the advantages inherent in each of these algorithms. We here present phoneme recognition results for this hybrid system which show that the high discriminant power of LVQ can be used in an HMM framework extendible to word or sentence recognition of continuous speech.

2. HMMs with an LVQ Codebook

2.1. Review of LVQ2

LVQ, recently developed by Teuvo Kohonen [1, 2], is an algorithm for vector quantization that is very closely related to Kohonen's work on self-organizing feature maps [1].

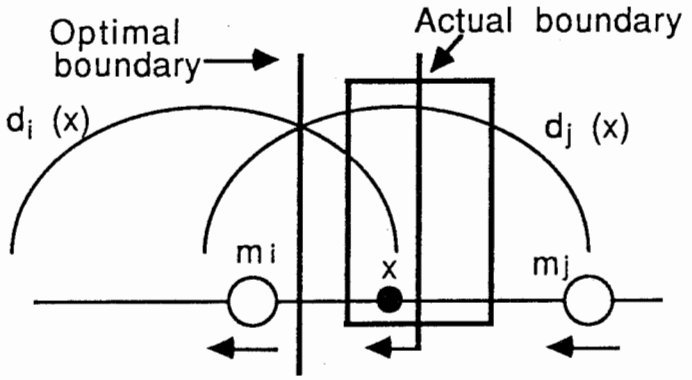
The goal of LVQ is to use reference vectors to divide the vector space in a pattern classification problem with decision lines that will correspond as closely as possible to the optimal Bayes decision rule. LVQ was shown in [3, 4] to yield very high pho-

neme classification rates even for a small number of reference vectors. We here describe the algorithm.

There are currently two versions of LVQ; here attention is focussed on LVQ2 [2]. In LVQ2, each category to be learned is assigned a number of reference vectors, each of which has the same number of dimensions as the input vectors of the categories. In the recognition stage, an unknown input vector will be categorized by finding the reference vector that is closest to that input vector. The category that the reference vector belongs to will be given as the categorization of the unknown input vector. In the learning phase, then, LVQ2 attempts to adapt the positions of the reference vectors such that each input vector has a reference vector of the right category as its closest reference vector.

Kohonen, in his formulation of the LVQ2 algorithm, is particularly concerned with the problem of what to do when the class distributions overlap. If there is overlap, it is impossible to separate the classes perfectly; the task becomes that of finding the decision line that will minimize the number of misclassifications. This will be achieved by a decision line at the place where the class distributions cross. Ideally, a neural network should generate decision lines that approximate this optimal line. This is the motivation for LVQ2.

Figure 1 helps to illustrate vector adaptation in LVQ2, for a simplified one-dimensional situation. For a given training vector x, three conditions must be met for learning to occur: 1) the nearest class must be incorrect; 2) the next-nearest class (found by searching the reference vectors in the remaining classes) must be correct; 3) the training vector must fall inside a small, symmetric window defined around the



x: training vector belonging to class j
m_i ,m_j : reference vectors of class i,j
d_i ,dj : distributions of classes i,j

Fig.1 LVQ2 in 1 dimension

midpoint of m_i and m_j-- this midpoint ("mid-hyperplane") being the decision boundary effected by the two vectors. If these conditions are met, the incorrect reference vector is moved further away from the input, while the correct reference vector is moved closer, according to:

$$m_i(t+1) = m_i(t) - \alpha(t)\,(x(t) - m_i(t))$$
$$m_j(t+1) = m_j(t) + \alpha(t)\,(x(t) - m_j(t))$$

where x is a training vector belonging to class j, m_i is the reference vector for the incorrect category, m_j is the reference vector for the correct category, and $\alpha(t)$ is a monotonically decreasing function of time.

It can be seen that this scheme assures that the decision line between the two vectors will eventually be very close to the optimal Bayes boundary given the probability distributions of the categories (in Fig. 1, $d_i(x)$ and $d_j(x)$), at the place where the distributions cross.

2.2. Review of Shift-Tolerant LVQ

The Shift-Tolerant LVQ phoneme recognition architecture referred to here was described and evaluated in [3], and illustrated here in Figure 2. In this architecture, each phoneme to be recognized is assigned a number of reference vectors. For the purpose of achieving a degree of tolerance to temporal shifts in the phoneme tokens (represented in [3] as 15 frames of 16 FFT coefficients), a 7-frame time window is defined, and shifted over the phoneme token, one frame at a time. Each position of this window yields an input vector of 112 dimensions, which is first used for K-means generation of an initial set of reference vectors, and then for LVQ2 training. The K-means initialization here is performed for just one category at a time, using only the tokens of that category. In this paper, we refer to this K-means procedure as "K-means each." This procedure was shown in [4] to be more effective as an initialization procedure for the Shift-Tolerant LVQ architecture than the more usual procedure, where K-means clustering is performed for all categories at the same time.

In the recognition phase of this architecture, the time window is shifted as before, and for each position of this window over the token, a simple measure of activation is calculated for each phoneme category. When the window has been fully shifted over the token, the activations over time are summed to yield final activations for each phoneme. The phoneme category with the highest activation is chosen as the categorization of the token.

This architecture is described in greater detail in [3].

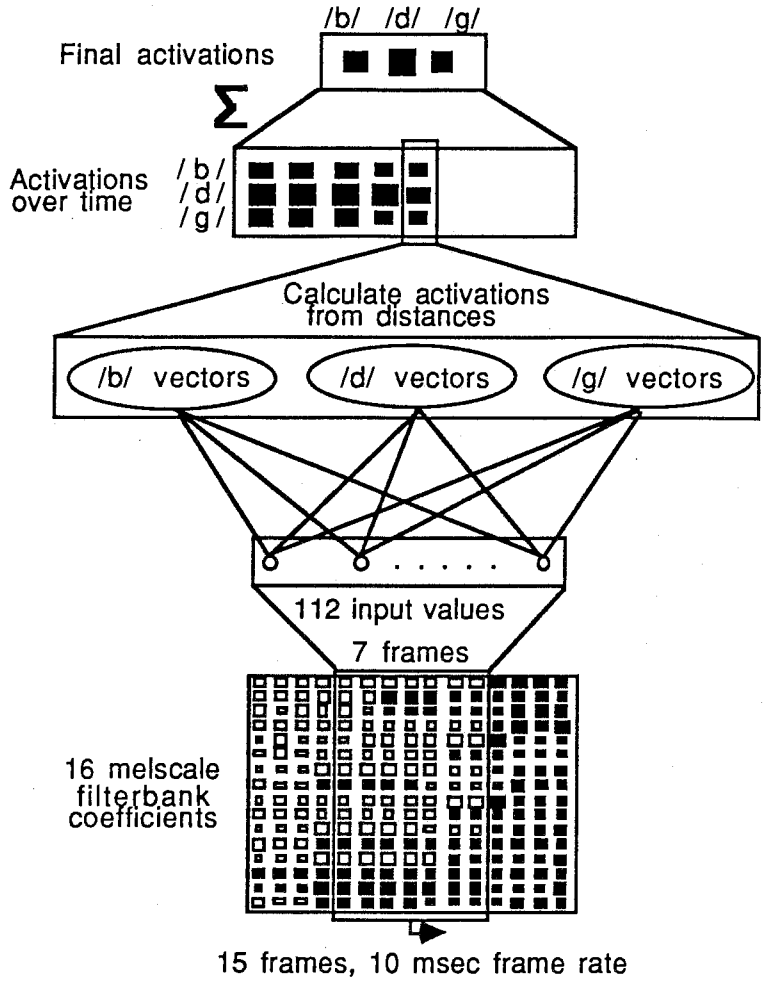


Fig.2 LVQ system architecture

2.3. LVQ-HMM System Architecture

Figure 3 shows a block diagram of a conventional HMM. In this model, the codebook is generated by the well-known K-means clustering algorithm. We will refer to this model VQ-HMM. On the other hand, Figure 4 shows the new model we are proposing here. In this model, the codebook consists of highly discriminant reference vectors generated by LVQ training. We will refer to this model as LVQ-HMM.

Figure 5 shows a block diagram of phoneme recognition using LVQ-HMM. In initializing the codebook, we first cluster all the feature vectors of the training samples, and generate a predetermined number of reference vectors. Next this codebook is trained using LVQ2 in order to provide the codebook with high classification ability. As in the usual HMM method, this codebook is used to convert all the training samples into a code sequence which is then used to estimate the parameters of each HMM. To evaluate the resulting phoneme models on unknown data, test samples were also converted to a sequence of codes.

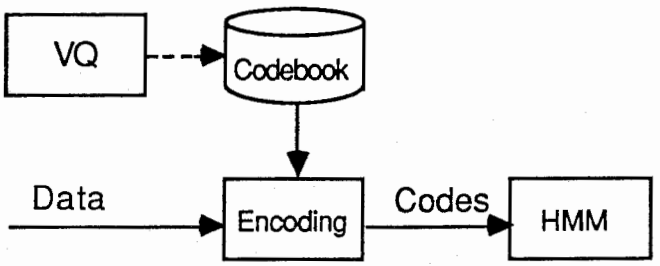


Fig.3 The concept of VQ-HMM

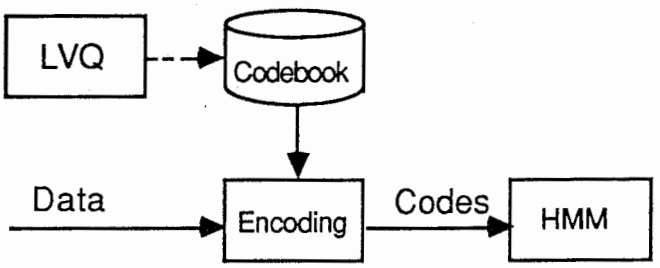


Fig.4 The concept of LVQ-HMM

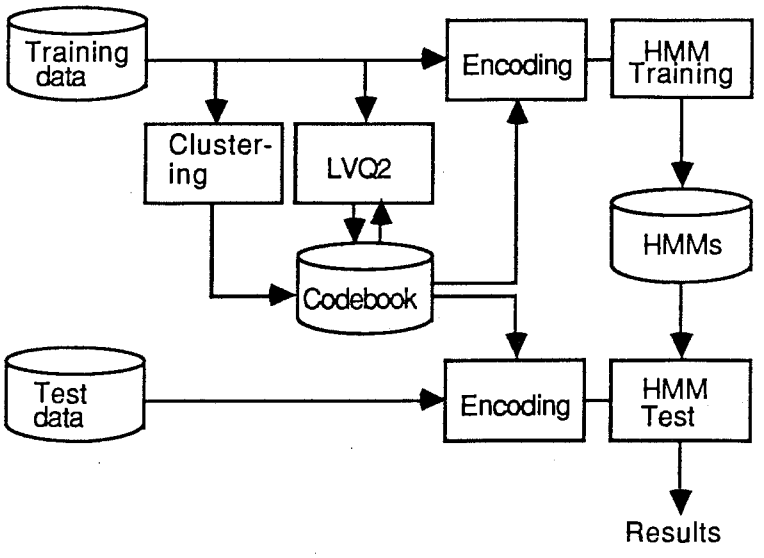


Fig.5 Block diagram of phoneme recognition using LVQ-HMM

2.4 Phone Models

There have been numerous studies, for instance [12], concerning the architecture that should be used for phone models. Here we summarize some conclusions.

1) There should be at least 3 states with loops per model.

2) The last state should not have a loop.

3) There is usually little difference in recognition rates between models that used tied arcs and models that don't.

Taking these considerations into account, we used the phone model structure shown in Fig. 6. In this figure, a_{ij} is the transition probability from state i to state j; and b_{ik} is the output probability of code k when there is a transition from state i.

3. Phoneme Recognition Experiments

The purpose of the following experiments was to evaluate the LVQ-HMM hybrid model proposed here. We performed two main comparisons:

1) Comparison of the new algorithm with Shift-Tolerant LVQ [3] and TDNN [5, 6]. In this experiment, phoneme recognition experiments were performed using the same data sets as used in TDNN and Shift Tolerant LVQ, for 7 Japanese phonemic classes.

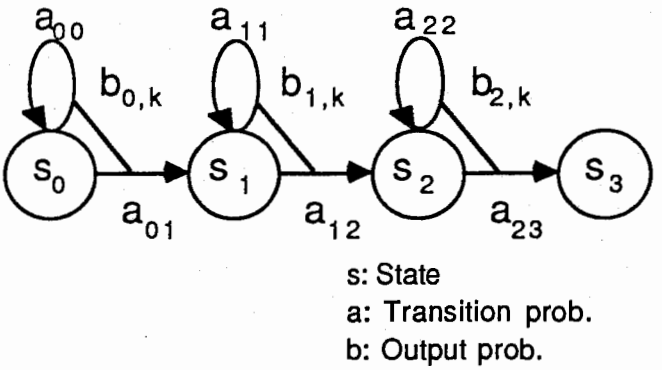


Fig.6 Phoneme model structure. A phoneme model consists of a left-to-right four state HMM. It is assumed that all the output probabilities associated with one state are equal.

2) Comparison of the LVQ-HMM hybrid with HMMs using conventional K-means generated codebooks. Phoneme recognition experiments were performed using phoneme tokens for 25 phoneme categories. Instead of using a fixed frame length for the data tokens, as in comparison (1) above, we here used tokens with a variable frame length that depended on the duration of each phoneme sample in the database.

3.1. HMM Training

The a_{ij} transition probabilities of the phoneme HMMs are all initialized to have equal values. The initial values of the observation probabilities b_{ik} are set, for each code k, at the number of observations of the code k, divided by the number of observations of all codes. The Baum-Welch algorithm, based upon maximum likelihood estimation, is used to train the HMMs. It is known that in this parameter estimation for phoneme models, convergence is attained after 5 iterations [12]; we set the number of iterations at 7. In order to prevent the output probabilities from becoming zero, should the number of training samples be too small, smoothing was performed to keep the parameters larger than 10^{-6}.

3.2 Speech Data

The tokens we used in all our experiments are drawn from a database of about 5240 common Japanese words, uttered in a sound-proof booth by a male professional announcer. The database was split into a training set and a testing set of 2620 utterances each, from which the phoneme tokens were then extracted using manually selected

acoustic-phonetic labels. The particular method of extraction is what differentiates the two data sets we used, which we now describe individually.

3.2.1. First Data Set

This data set is identical to the one used in Shift-Tolerant LVQ [3] and TDNN [6]. The phoneme tokens-- for 23 phoneme categories, nearly covering the entire phoneme set for Japanese-- were of fixed length and consisted of 15 time frames of 16 mel-scale spectrum channels, with a frame rate of 10 msec. Input speech was sampled at 12 kHz, hamming windowed, and a 256-point FFT computed every 5 msec. Melscale coefficients were generated from the power spectrum and coefficients adjacent in time were collapsed, yielding an overall frame rate of 10 msec. The coefficients were then normalized between -1.0 and 1.0 with the average at 0.0. These coefficients are displayed in Figure 2 as black or white squares of varying sizes, size representing magnitude, black for positive values, white for negative values. It should be noted that the token extraction rules used here were also the same as for the Shift-Tolerant LVQ and TDNN experiments. For vowels, the center frame of each token was set at the center position of the vowel. For consonants, the center frame was set at the border between the consonant and the following vowel.

This first data set was split into training set and test set, each containing essentially the same number of samples as the training and test sets used in Shift-Tolerant LVQ and TDNN.

3.2.2 Second Data Set

The second data set is very similar to the first one, with the main difference being that here we used tokens with a variable frame length that depended on the duration of each phoneme sample in the database. Coefficients adjacent in time were not collapsed as in the first data set, and were normalized so that the average lay at 0.0 (without limiting coefficient values between -1.0 and 1.0). This normalization was performed during LVQ training, with the LVQ time window length determining the duration over which the normalization was performed. The second data set also differs from the first in that it includes tokens from 2 additional phone classes, /f/ and /j/, as well as tokens from some phoneme contexts that had been excluded from the first data set. Thus the second data set includes a greater variety of phonemic expression.

The second set too was split into a training set and a test set, of roughly equal sizes.

3.3. Preliminary Experiments on the /b/, /d/, /g/ task

In order to guide the two comparisons just mentioned, we first ran some preliminary experiments on the task of recognizing /b/, /d/, and /g/, for tokens taken from the first data set. Specifically, the target of these preliminary investigations was to understand the effectiveness of LVQ compared to K-means as the generator of the HMM codebook, for two different codebook sizes and for two methods of K-means clustering.

In these preliminary experiments, we also examined codebook distortion vs. classification rate. Whereas conventional HMM theory assumes that the codebook should minimize distortion, the LVQ-HMM hybrid we are proposing here suggests that a codebook should instead minimize the number of phoneme classification errors. This is an important point, which will be illustrated in these preliminary /b/, /d/, /g/ experiments.

The training set here consisted of a total of 682 samples (219 for /b/, 203 for /d/, and 260 for /g/), while the test set consisted of 658 samples in all (227 for /b/, 179 for /d/, and 252 for /g/).

3.3.1 HMM Codebook Design: K-means Each vs. K-means All

The two K-means clustering procedures we used were as follows. The first is here referred to as "K-means each," and consists in performing K-means on the training vectors of just one category at a time. This is a very fast process, yielding several codebooks-- one per category-- which are then grouped together into a single codebook. This contrasts with the more common method, here referred to as "K-means all," where the training vectors of all categories are considered at the same time.

To test these two methods for K-means clustering, we looked at two codebooks, each consisting of a total of 255 reference vectors, generated from phoneme speech segments that are stepped through in time, as described in [3]. One of these codebooks was generated using K-means each (85 reference vectors per category), the other using K-means all. Each of these codebooks was then used as the codebook for a /b/, /d/, /g/ HMM, which was then trained with the method described above.

Table 1 shows the classification rates on test data for these two HMMs. As shown in this table, the HMM with the K-means each codebook gives better classification performance here, with less than half the error rate of the HMM using the K-means all codebook. Thus, these results suggest that K-means each is better suited to the goal of maximizing classification performance. This is in line with the results of similar investigations of these two methods of K-means clustering, reported in [4].

Table 1 Recognition results on test tokens for a /b/, /d/, /g/ HMM system, for four different codebooks. K-means All refers to k-means performed on all categories at the same time; K-means Each refers to k-means performed on one category at

Codebook Design	Codebook Size	recognition rate (%)				VQ distortion
		b	d	g	ave.	
K-means All	255	89.9	99.4	97.2	95.3	1.959
K-means Each	255 (85/cat.)	98.7	97.2	97.6	97.9	2.004
K-means Each	30 (10/cat.)	70.5	91.6	85.7	82.1	3.839
K-means Each --> LVQ	30 (10/cat.)	98.7	100.	97.6	98.6	6.727

Also in line with the investigations in [4] is the result here that the HMM using the K-means each codebook has recognition rates that are more even across the categories than in the HMM using the K-means all codebook.

3.3.2 HMM Codebook Design: LVQ vs. K-means Each

Next, we investigated the efficiency of an HMM using an LVQ codebook compared to that of an HMM using a K-means each clustered codebook. Here the codebook size was much smaller than above, consisting of 10 reference vectors per category (for a total of 30 vectors in the /b/, /d/, /g/ task). This number was deemed sufficient since Shift-Tolerant LVQ applied to the /b/, /d/, /g/ task can achieve very high performance using this number of reference vectors [3].

The procedure here is first to create the K-means each codebook, which is used in one of the HMMs. The LVQ codebook is then created by using the K-means each codebook to initialize the LVQ reference vectors, and then training those reference vectors using LVQ2, as described in [3]. Training is performed so as to get as high a recognition rate as possible using the Shift-Tolerant LVQ recognition architecture. The resultant codebook is then used in the second HMM. Both HMMs are then trained in the manner described above.

The results on test data for these two HMMs are shown in Table 1. As can be seen, the HMM using the LVQ codebook has an error rate ten times smaller than that

of the HMM with the K-means codebook.

3.3.3 VQ Distortion vs. Classification Rate

The distortions for the 4 codebooks examined in the two preceding sections are shown in Table 1. The definition of distortion here is the usual mean-square error, using the same test samples as used to calculate recognition rates. As can be seen from Table 1, the correlation between distortion and classification rate is tenuous. In fact, the codebook with the highest performance, that generated using LVQ2, has the highest distortion. Furthermore, in the comparison between K-means each and K-means all, the higher performing K-means each yields a codebook with a greater distortion than that produced by K-means all.

3.4. Recognition Within 7 Consonant Clusters

Encouraged by these results for /b/, /d/ and /g/, we turned to additional phoneme classes: the unvoiced stops, /p/, /t/, /k/; the nasals /m/, /n/, and syllabic nasals; fricatives /s/, /sh/, /h/, /z/; affricates /ch/ and /ts/; liquids and glides /r/, /w/, /y/; and the vowels /a/, /i/, /u/, /e/, /o/. Together, these constitute nearly the entire phoneme set for Japanese. The size of the training sets for these classes was roughly the same as the size of the test sets, shown in Table 2. Note that the phoneme tokens here, as in the /b/, /d/, /g/ experiments above, were taken from the first data set, i.e., had a fixed length of 15 frames; this is the same data set as used in Shift-Tolerant LVQ and TDNN.

Here we compared the performances of three systems: (1) HMMs using an LVQ trained codebook, (2) Shift-Tolerant LVQ, and (3) TDNN. Note that the reference vectors in (1) and (2) are identical, the result of the same Shift-Tolerant LVQ training. In the case of (1), the LVQ reference vectors are used as the codebook to an HMM, trained as above; in the case of (2), the reference vectors are used for the very simple generation and summation of activations in the Shift-Tolerant recognition architecture, described in [3]. The number of reference vectors varied with each phonemic class examined, from 10 to 30 reference vectors per class. (3) Refers to the TDNN system described and evaluated in [6].

The recognition rates for these three systems are shown in Table 2. As can be seen, the LVQ-HMM hybrid attained a recognition performance as high as that of Shift-Tolerant LVQ and TDNN.

Table 2 Recognition results for 7 phonemic classes

task	LVQ+HMM			LVQ	TDNN
	#errors/ #tokens	correct%	ave.%	ave.%	ave.%
b	3/227	98.7	98.6	99.2	99.0
d	0/179	100.			
g	6/252	97.6			
p	7/ 15	53.3	98.4	98.9	98.7
t	2/440	99.5			
k	6/500	98.8			
m	7/481	98.5	98.5	98.8	96.6
n	6/265	97.7			
N	5/488	99.0			
s	5/538	99.1	99.3	99.4	99.3
sh	0/316	100.			
h	0/207	100.			
z	3/115	97.4			
ch	0/123	100.	100.	100.	100.
ts	0/177	100.			
r	0/722	100.	99.9	99.6	99.9
w	0/ 78	100.			
y	1/174	99.4			
a	0/600	100.	98.9	99.1	98.6
i	2/600	99.7			
u	21/600	96.5			
e	7/600	98.8			
o	2/600	99.7			

3.5. Recognition of All Consonants

We next turned to the more difficult task of recognizing all the above consonants taken together, not just within small phonemic classes. Here we are only comparing the LVQ-HMM hybrid with Shift-Tolerant LVQ. The data set here is still the first data set, of fixed length tokens. Twenty five reference vectors were assigned to each category, and LVQ training was performed as above. As above, the resulting reference vectors were used in an HMM framework for training and recognition; the same reference vectors were also used in the Shift-Tolerant LVQ recognition architecture. The performances on test data of these two systems are shown in Table 3. Once again, it can be seen that the LVQ-HMM hybrid performs as highly as Shift-Tolerant LVQ.

Table 3 Recognition results of 18 consonants

task	LVQ+HMM			LVQ
	#errors/ #tokens	correct%	ave.%	ave.%
b	2/227	99.1		
d	2/179	98.9		
g	14/252	94.4		
p	7/ 15	53.3		
t	9/440	98.0		
k	18/1163	98.5		
m	7/481	98.5		
n	7/265	97.4		
N	14/488	97.1	97.4	97.7
s	16/538	97.0		
sh	3/316	99.1		
h	6/207	97.1		
z	6/115	94.8		
ch	6/123	95.1		
ts	10/177	94.4		
r	14/722	98.1		
w	6/ 78	92.3		
y	6/174	96.6		

3.6. Phoneme Recognition Using Tokens of Variable Length

We then turned to the second data set described above. The phoneme tokens in this data set are taken from a greater variety of phonemic contexts than in the first data set; in addition the second data set contains tokens from two additional phonemes, /f/ and /j/, giving a total of 25 phonemes.

It was very simple to modify the Shift-Tolerant LVQ architecture to deal with the variable token length. At the first level, where LVQ2 training is performed (using reference vectors initialized by K-means each), the 7-frame time window was simply shifted over the length of the whole token, no longer limited at 15 frames. LVQ2 training was performed for each position of this time window over the token, as in the fixed frame length architecture. At the second level of the architecture, where activations are calculated and summed, the history of activations was simply extended or shortened depending on the token length. The calculation of final activations was obtained from the same process of summation as before.

The HMMs we examined here are trained and tested as above, with the excep-

Table 4 Recognition results for 25 consonant tokens of variable length. K-means All refers to k-means performed on all categories at the same time; K-means Each refers to k-means performed on one category at a ti

Recognition System	K-means Procedure	Codebook Size	
		250	625
VQ-HMM	All	92.8	96.1
VQ-HMM	Each	94.7	96.9
LVQ-HMM	Each	97.2	98.0
Shift Tolerant LVQ	Each	95.3	97.5

tion that the tokens used for training are of variable length.

Here too we performed a number of experiments. We examined two codebook sizes: 250 reference vectors in all (10 per category for K-means each) and 625 reference vectors in all (25 per category for K-means each). Our focus here was to compare VQ-HMM, LVQ-HMM, and Shift-Tolerant LVQ, for these two codebook sizes. In addition, for VQ-HMM, we compared K-means each and K-means all.

Table 4 shows the results for these systems. These results confirm the finding of the preliminary investigations of section 3.2, that K-means each seems to provide the HMMs with better discriminant ability than K-means all. We also see that Shift-Tolerant LVQ by itself performs better than VQ-HMM, and that the LVQ-HMM hybrid performs significantly better than the VQ-HMM. For the large codebook, the hybrid model's error rate is 1.5 ~ 2 times lower than that of VQ-HMM; for the small codebook, 2 ~ 3 times lower.

4. Discussion

The above experiments suggest a rather different approach to codebook design, concerned not so much with reducing spectral distortion, but rather with providing information as to the phonemic identity of the speech segments under consideration. Table 1 illustrates this point. For codebooks of the same size, we see that there is in fact an inverse relationship between distortion and the HMM's recognition rate. It

makes sense that K-means all would be better at reducing distortion than K-means each, as it takes the whole data set into account. It is not so clear why K-means each yields a configuration better suited to reducing the number of misclassifications. It could be that K-means each is better at describing the probability distribution of each category, especially at the category boundaries. This explanation is consistent with the motivation for LVQ, which is to pay very close attention to the category boundaries, for the purpose of obtaining high classification performance. When comparing the LVQ codebook with the K-means codebooks, the inverse relationship between distortion and recognition rate is readily understood. LVQ starts from a K-means configuration, where distortion is minimal, and adjusts that configuration for the purpose of reducing the number of misclassifications. Clearly the resulting configuration will no longer minimize distortion, but the classification rate will be better than that of the original configuration.

In Table 1, we see how using an LVQ codebook can provide an HMM with a better recognition rate than a K-means codebook that is 10 times larger. This reduction in codebook size is a general feature of LVQ which can benefit several speech recognition approaches, such as multi-template matching and Shift-Tolerant LVQ. The main advantage is a large reduction in computation time (the search for the closest reference vector can be done in less time as there are fewer reference vectors), as well as an improvement in performance. In the hybrid LVQ-HMM system at hand, there is the additional advantage of making HMM parameter estimation easier. The number of observation probabilities to be estimated is decreased, and this in turn reduces the quantity of training data necessary to estimate these parameters. In effect, the overall size of the HMM recognition system, including both codebook and parameters, is very significantly reduced. Thus a small LVQ codebook, compared to a much larger K-means codebook, can provide an HMM with substantial gains in memory size, speed and performance.

From Tables 2 , 3 and 4, we see that the phoneme recognition performance for LVQ-HMM is as high as that of Shift-Tolerant LVQ. The advantage of the hybrid LVQ-HMM system proposed here is that it can very easily be extended to models for longer utterances, such as words, phrases and sentences. Thus, even though the results we present here only concern phoneme recognition, the fact that the LVQ-HMM hybrid performs phoneme recognition in an HMM framework establishes the crucial link between the high discriminant power of the Shift-Tolerant LVQ system and the ability HMMs have to concatenate small utterance models into long utterance models.

The hybrid presented here is just one of several possibilities for using LVQ in combination with HMMs. In addition, LVQ is also open to integration with other speech recognition techniques, such as DTW.

5. Summary

The motivation of our study was to integrate the high discriminant power of the LVQ classifier into an HMM framework extendible to long utterance models. Our findings were as follows.

1) Codebook distortion is not a good index of HMM classification performance. Codebooks generated by performing K-means on just one category's data at a time resulted in higher spectral distortion, but better HMM recognition performance, than the more conventional method of performing K-means on data from all categories at the same time. These effects are even more pronounced when comparing LVQ-generated codebooks used in an HMM framework with codebooks generated by either K-means method.

2) Applying LVQ to a small K-means generated codebook resulted in a ten-fold reduction of the HMM error rate for the /b/, /d/, /g/ task.

3) Applying the LVQ-HMM hybrid system to a variety of phonemic recognition tasks, including tasks where the phoneme tokens were of variable length, yielded results that were as high as those for the Shift-Tolerant LVQ system.

Given the above findings, it seems the LVQ-HMM hybrid proposed here successfully adopts a new approach to discrete HMM speech recognition, using a codebook whose purpose is to classify phonemes correctly rather than to minimize spectral distortion.

Acknowledgments

The authors wish to thank Toshiyuki Hanazawa in the ATR Interpreting Telephony Research Laboratories for helping us with HMM software. Thanks are also due to Dr. Kiyohiro Shikano, also in the ATR Interpreting Telephony Research Laboratories, for his valuable advice.

References

[1] T. Kohonen, *Self-organization and Associative Memory* (2nd Ed.), pp. 199-202, Springer, Berlin-Heidelberg-New York-Tokyo, 1988.

[2] T. Kohonen, G. Barna and R. Chrisley; "Statistical Pattern Recognition with Neural Networks: Benchmarking Studies," IEEE, Proc. of ICNN, Vol. I, pp. 61-68, July 1988.

[3] E. McDermott, S. Katagiri, "Shift-Invariant, Multi-Category Phoneme Recognition using Kohonen's LVQ2," in Proc. Int. Conf. Acoust., Speech, Signal Processing, Glasgow, U.K., 1989, pp. 81-84.

[4] M. Yokota, S. Katagiri and E. McDermott; "Learning in an LVQ-Based Phoneme Recognition System," Inst. Elec. Inf. Com. Eng. of Jpn. IEICE Technical Report, SP88-104, pp. 65-72, 1988 (in Japanese)

[5] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang, "Phoneme Recognition Using Time-Delay Neural Networks," in IEEE Trans. Acoust. Speech, Signal Processing, vol. ASSP-37, pp. 328-339, 1989.

[6] A. Waibel, H. Sawai, K. Shikano, "Consonant Recognition by Modular Construction of Large Phonemic Time-Delay Neural Networks," in Proc. Int. Conf. Acoust., Speech, Signal Processing, Glasgow, U.K., 1989, pp. 112-115..

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

[8] H. Bourlard and C.J. Wellekens, "Links between Markov Models and Multilayer Perceptrons," in Proc. NIPS Conference, Denver, CO, 1988.

[9] K-F. Lee and H-W. Hon; "Large-Vocabulary Speaker Independent Continuous Speech Recognition Using HMM," in Proc. Int. Conf. Acoust., Speech, Signal Processing, New York, NY, 1988, pp. 123-126.

[10] P. Brown., *The Acoustic-Modeling Problem in Automatic Speech Recognition*, PhD dissertation, Computer Science Department, Carnegie Mellon University, 1987.

[11] L.R. Bahl, P.F. Brown, P.V. de Souza, and K.L. Mercer, "Maximum Mutual Information Estimation of Hidden Markov Parameters for Speech Recognition," in Proc. Int. Conf. Acoust., Speech, Signal Processing, Tokyo, Japan, 1986, pp. 49-52.

[12] T. Hanazawa, T. Kawabata, K. Shikano, "Recognition of Japanese voiced stops using Hidden Markov Models," Inst. Elec. Inf. Com. Eng. of Jpn. IEICE Technical Report, SP87-98, pp. 7-12, 1987. (in Japanese)

[13] G. Doddington, "Phonetically Sensitive Discriminants for Improved Speech Recognition," in Proc. Int. Conf. Acoust., Speech, Signal Processing, Glasgow, U.K., 1989, pp.556-559.

[14] L.R. Bahl, et. al., "Large Vocabulary Natural Language Continuous Speech Recognition," in Proc. Int. Conf. Acoust., Speech, Signal Processing, Glasgow, U.K., 1989, pp.465-467

[15] D.J. Burr, "Speech Recognition Experiments with Perceptrons", in Neural Information Processing Systems (E. ANderson Ed.), pp. 144-153, American Institute of Physics

[16] R.P. Lippmann and B. Gold, "Neural Net Classifiers Useful for Speech Recognition," Proc. of International Conference on Neural Networks, IEEE, San Diego, June 1987.

[17] H. Sakoe, R. Isotani and K. Iso, "Speaker-Independent Word Recognition Using Dynamic Programming Neural Networks," in Proc. Int. Conf. Acoust., Speech, Signal Processing, Glasgow, U.K., 1989, pp. 29-32.

[18] L-Y. Bottou, "Reconnaissance de la Parole par Reseaux multi-couches." Proc. of Neuro-Nimes 88, Novermber 1988.