ROBUST ISOLATED SPEECH RECOGNITION USING IDEAL BINARY MASKS

Seliz Karadogan, Jan Larsen, DTU Informatics, Richard Petersens Plads, B321, DK-2800 Kongens Lyngby

Jesper Boldt, Michael Syskind Petersen, Oticon, Kongebakken 10, 2765 Smørum

This is supplementary material the submitted conference paper of same title for ICASSP 2010. We do not repeat what is already written in the paper, i.e. this material may not be sufficient for the reader to fully understand this document. This material includes some additional figures that might be interest of the readers of the paper and the optimization process for the parameters of the project used for the experiments the results of which are given in the paper.

We work on a speaker-independent isolated digit recognition using ideal binary mask (IBM). An IBM is 1 when the target is greater than the noise for a local criteria(LC) defined and zero elsewhere as in Equation 1 where T(t, f) and N(t, f) denote the target and noise time-frequency magnitudes respectively .

$$IBM(t,f) = \begin{cases} 1, & \text{if } T(t,f) - N(t,f) > LC \\ 0, & \text{otherwise} \end{cases} \qquad (1)$$

11 digits are recognized from ‘zero’ to ‘nine’ including the digit ‘oh’. The IBMs for 11 digits can be seen in Figure 1.

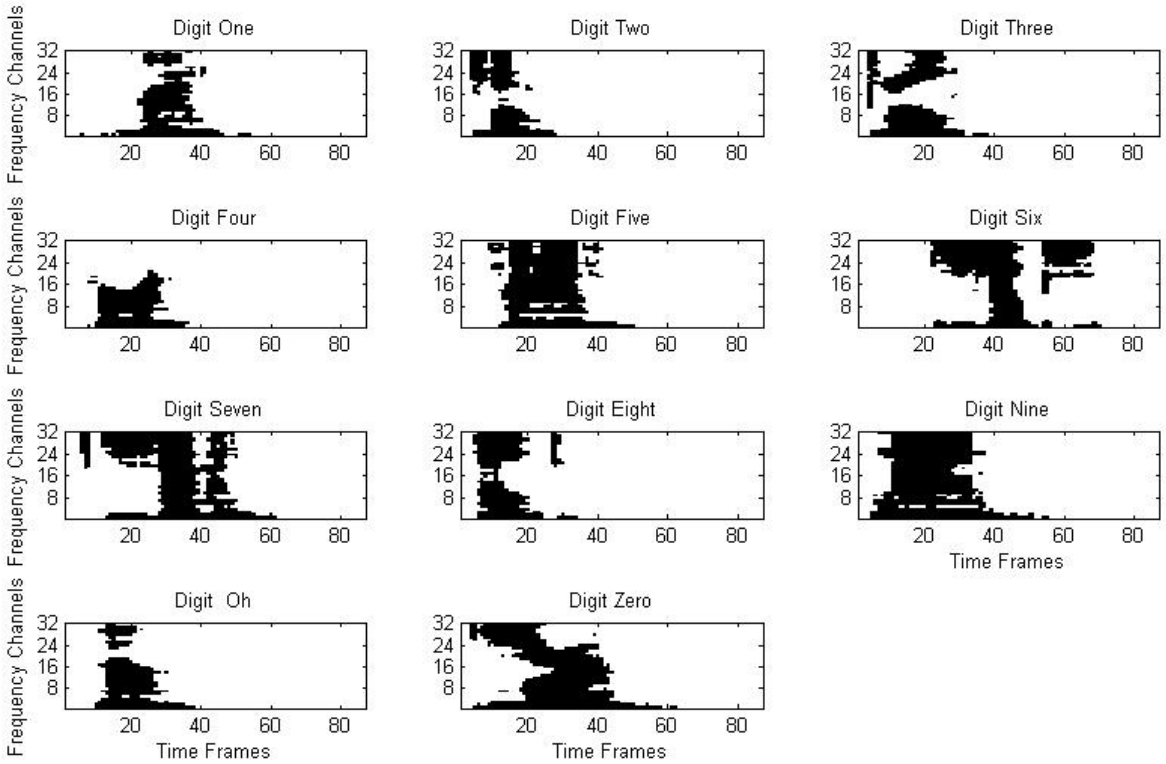

Figure 1: IBMs for 11 digits for Speech Shaped Noise(SSN) with SNR = 6 dB dB ; LC = 0 dB

The Hidden Markov Models (HMM) are trained with features obtained from IBMs. For a digit sample, IBMs are obtained for different SNR values in the range of [-2dB,16dB] that results in IBMs with different densities. The IBMs for the digit 'three' can be seen In Figure 2.

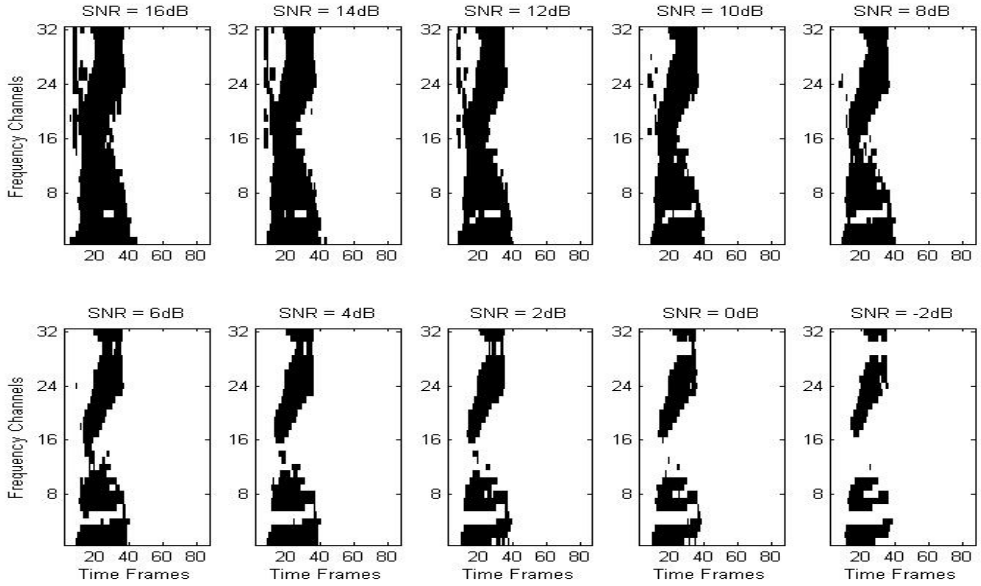

Figure 2: IBMs for digit three with SSN for different SNR values used for training with LC = 0 dB

Binary masks for noisy speech are also obtained. Figure 3, 4 and 5 show the binary masks with different SNR values for car, bottle and cafe noises respectively where the reference noise signal is SSN again.

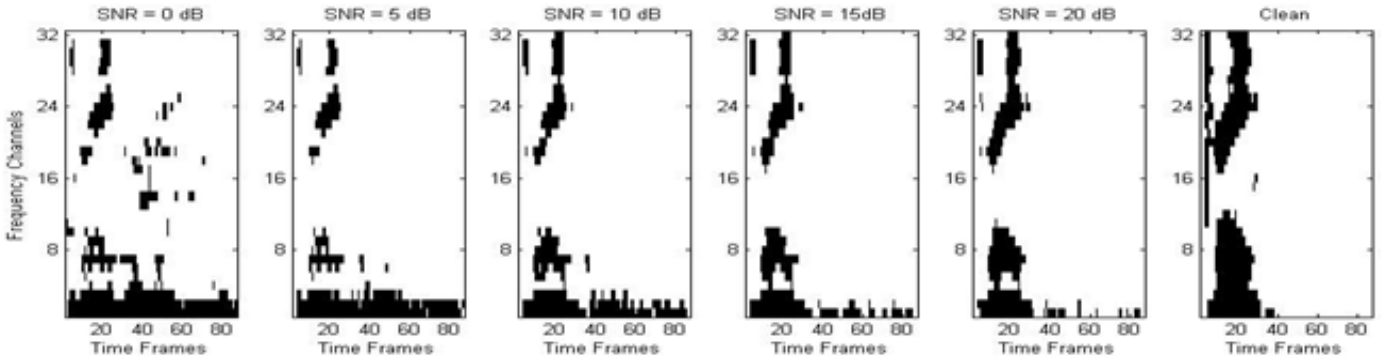

Figure 3: IBMs for digit three for car noise with different SNR values and the LC values adjusted for best performance

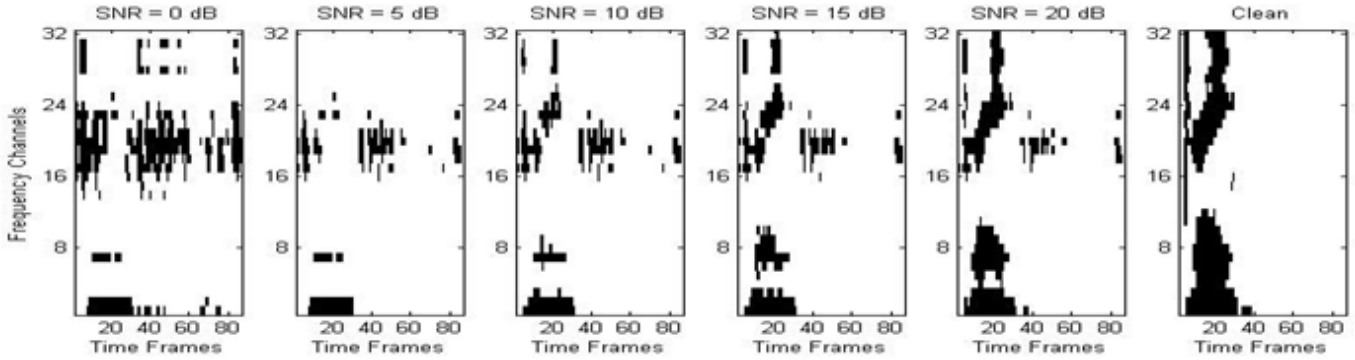

Figure 4: IBMs for digit three for bottle noise with different SNR values and the LC values adjusted for best performance

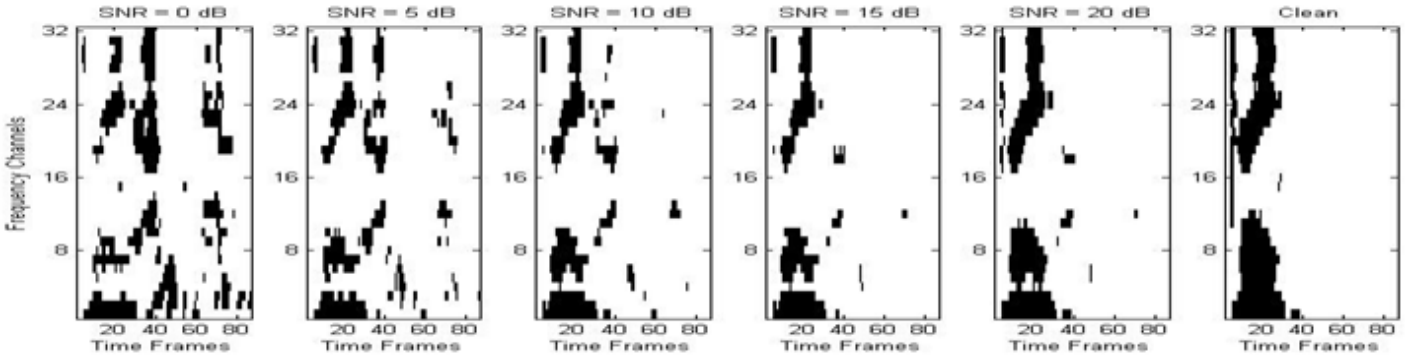

Figure 5: IBMs for digit three for cafe noise with different SNR values and the LC values adjusted for best performance

OPTIMIZATION OF THE VARIABLES:

First, we needed to find the optimal values for the variables in our project. The main variables to be optimized are:

- **Vector Size:** The number of columns of a mask to be stacked as input to K-means
- **K:** Number of vectors in the codebook resulted from K-means algorithm
- **N:** Number of states in HMM
- **FC:** Number of frequency channels of an IBM
- **WL:** Window length, the length of time slots of an IBM in milliseconds

Optimizing those variables is not a trivial task since they constitute a five dimensional system (in fact the number of available training data can be considered as the sixth dimension) where changing the value of one of them might affect the others. We tried to keep that task as simple as possible by keeping some of them constant. SNR, local criteria LC and data numbers used are kept constant for this part unless otherwise stated. Their values can be seen in Table 1 the values of other variables that are changed during experiments can be read from the caption parts of every figure.

Table 1: The variables kept constant through the optimization process

Training data number	174 (74 men, 100 women)
Verification data number	87(37 men, 50 women)
SNR	0 dB
LC	-8 dB

First, we start with the optimal vector size. Since the size of the codebook K can affect the recognition results for different number of vector sizes, we first investigated this effect. In Figure 6, we see the impact of changing the value of K on the overall recognition results for different vector sizes. We observe similar curves for all vector sizes. As K increases the recognition rate increases as well, until the value of K=48. We could call K=48 as the break-up point since after that point the rate changes inconsiderably. We conclude that the optimal number for K should be greater than or equal to 48. Then we look for CPU time passed for obtaining those results. CPU time passed increases as the value of K increases as seen In Figure 7. This result is quite expected since as K increases, in every iteration of K-means, vectors are matched to a larger number of center points. Considering these results, we conclude that the optimal number for the size of the codebook K is 48 for 174 IBMs used for training.

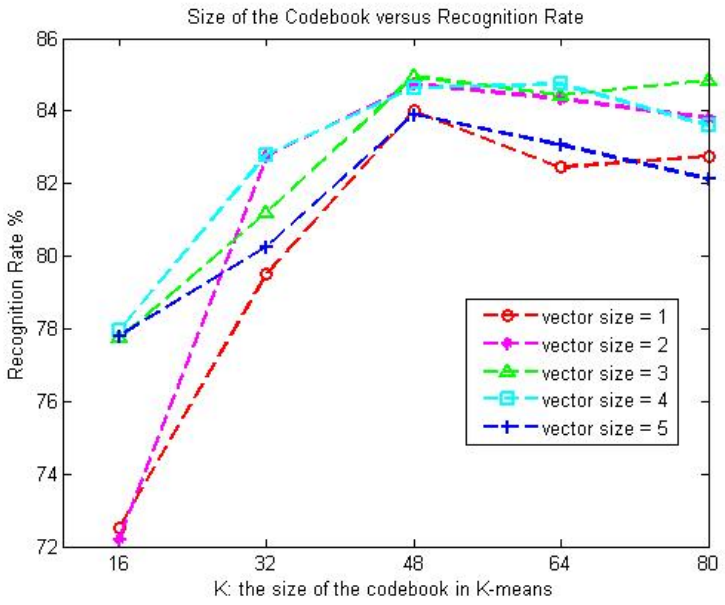

Figure 6: The recognition rates for different codebook sizes where N = 6, FC = 32, WL=20ms

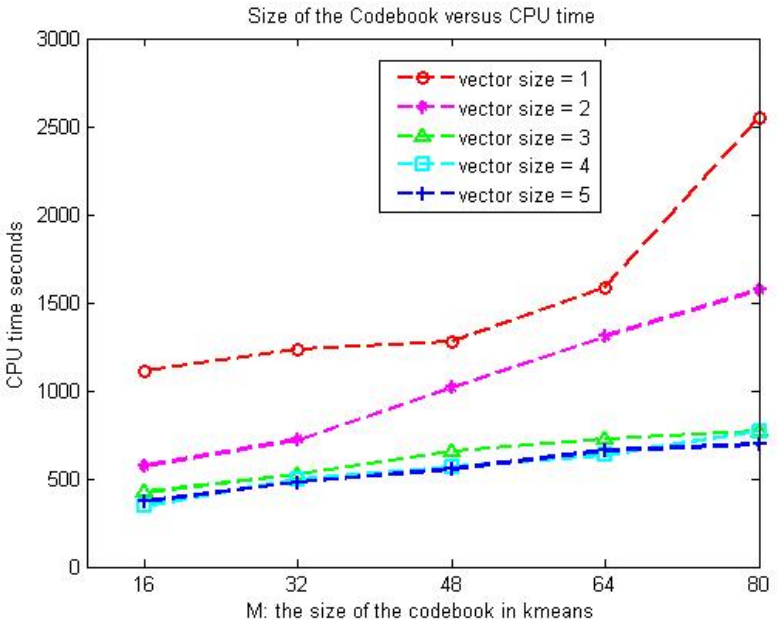

Figure 7: The CPU time passed for different codebook sizes where N = 6, FC = 32, WL=20ms

We continue with the search for optimal number of vector size with K being 48. Figure 8 shows us how the recognition results and CPU time passed change with changing the vector size. It is seen that as the vector size increases, the CPU time passed decreases due to the fact that the number of vectors to be matched to the codebook decreases. However, the decrease of CPU time is much more slowly after the vector size of 3. In that figure, we also see that the recognition is maximal at vector size 3 which makes it the optimal number. It should be pointed that the recognition rate is maximal for a vector size and decreases either by increasing or lowering the vector size. As the vector size is increased, while the similarities between same digits increase, the differences between different digits decrease and vice versa. Thus, we conclude that the similarities and differences are optimum at vector size of 3.

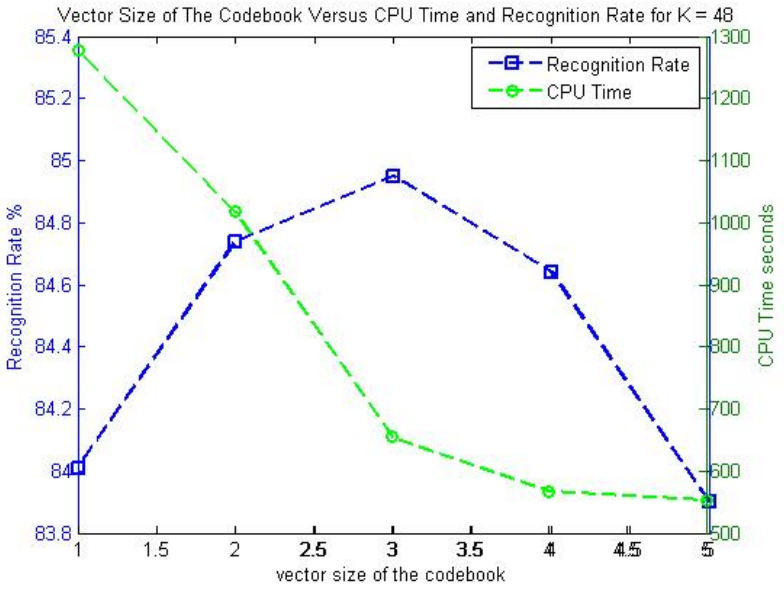

Figure 8: The recognition rates and CPU times passed for different vector sizes where K=48, N = 6, FC = 32

With K of 48, and vector size of 3, we continue with optimal number of states. The change in the recognition results with the number of states can be observed in Figure 9. The experiments have been done with 12 different numbers of states from 3, the minimum number of states for a left-right HMM, to 14. We see that the recognition is maximal as 88% with state number 10. Then, we conclude that the optimal number of states is 10.

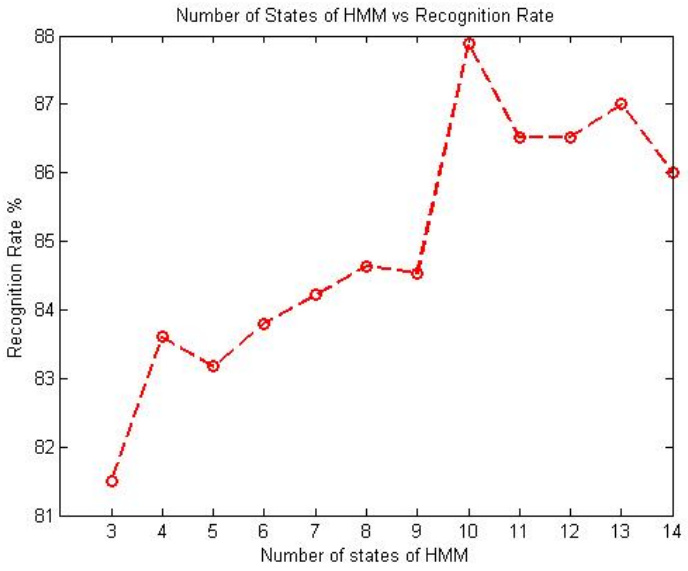

Figure 9: The recognition rates for different numbers of states of HMM where vector size=3, K=48, FC = 32, WL=20ms

The window length and the frequency channels that are used within gammatone filtering part, could be considered as the parameters specifying the resolution of the IBM. We might want to increase this resolution so that we have a better recognition performance, but computational load is also to be minimized. Then, these two variables also should be optimized as we did for other variables. Figure 10 gives the recognition results and CPU time passed for three different window lengths that are 10,20 and 40 ms. As we increase the window length, the size of the IBMs decreases resulting in a decrease of the CPU time passed. While recognition results for window lengths 10ms and 20ms do not differ too much, it rapidly decreases for 40ms making it a weak candidate for the optimal value. In addition, the difference between

CPU time passed for 20ms and 40ms is not so high. These results lead us that the optimal window length is 20 ms owing both to the high recognition result and acceptable computational load.

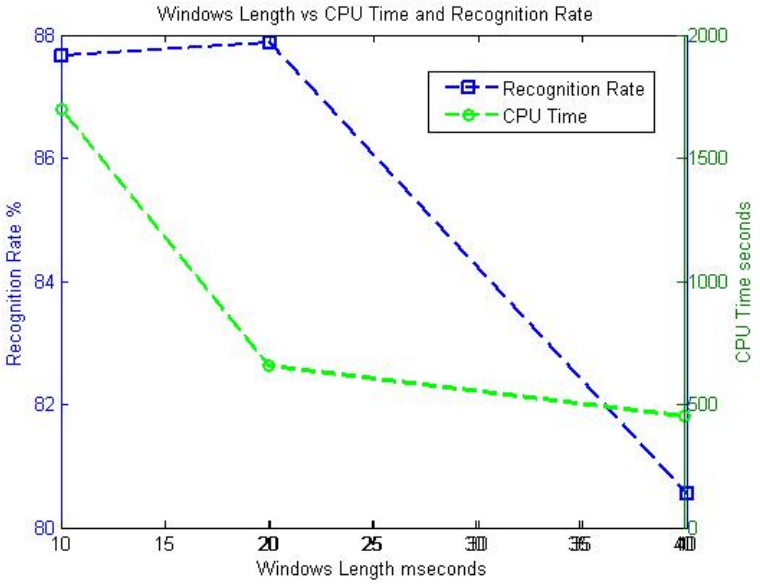

Figure 10: The recognition rates and CPU Time Passed for different window lengths where vector size=3, K=48, FC = 32, WL=20ms

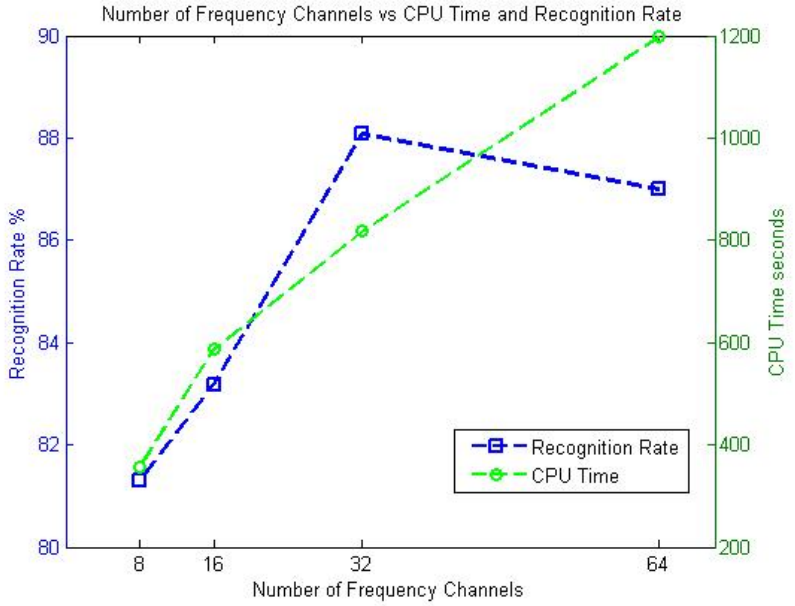

Figure 11: The recognition rates and CPU Time Passed for different number of frequency bands where vector size=3, K=48, FC = 32, WL=20ms

It is expected that increasing the number of frequency bands(FB) results in higher CPU time passed and this can be easily observed from Figure 11. In this figure, one can also see that the rate of change of CPU time is almost constant. And since we want to keep the computational load as low as possible, we try to find the minimum number of FB giving the best recognition results. It is seen that, the recognition rates seem to converge at FB of 32 which makes it the optimal number for FB. One may find the results in the Figure 10 and Figure 11 conflicted in the way that increasing the resolution, WL from 20ms to 10 ms and FB from 32 to 64, does not bring about higher recognition results. However, it is not the case and has a very simple explanation. The number of vectors to be classified is increased by increasing the resolution of IBMs which eventually result in a higher optimal codebook size. Thus, we might have slightly better recognition results with WL 10 ms or FB 64 but with K being more than 48 which is the number being used for this experiment, however with much more computational load.

Another point to be studied is the effect of the number of training data on the size of the codebook K. Our training data number is limited to 174, but with IBMs with different LC numbers we could increase it. We found the optimal number of K with different training data from 50 to 1740 which can be seen in Figure 12. As seen, as the number of training data is increased with IBMs of many LC values, the optimal number for K increases. This is an expected result, since as the range of LC increases the diversity of IBMs increases.

Therefore, the diversity of vectors to be categorized into different classes increases. However, as the range of LC increases to a value that IBMs are close to all zeros and all ones, the diversity of vectors converges. Thus, it is expected that the optimal number of K converges with the increased number of training data. However, within the experiments, we used 1740 training data at most, which makes optimal K number as 256.

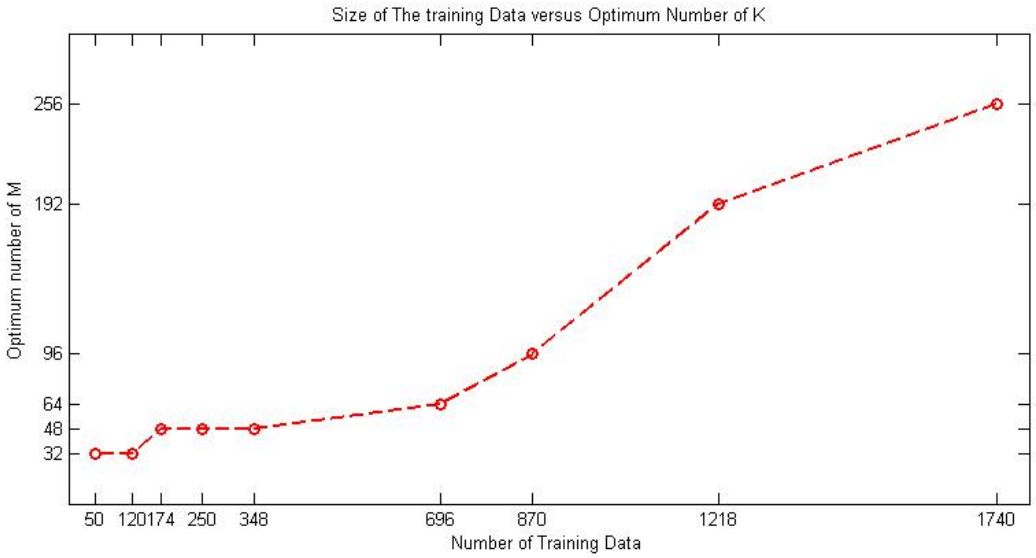

Figure 12: The optimal number of the codebook size K for different number of training data, to increase the training data IBMs with different LC numbers have been used

To conclude the part of the optimization of main variables, we give the optimal values for 174 training data in Table 2, with constants used given in Table 1 .

Table 2: The optimal values for the variables used through recognition process

Vector Size Number	3

Codebook Size K	256
Number of HMM States	10
Frequency Channel Number	32
Window Length	20 ms