19 Using SOM and LVQ for HMM Training

Mikko Kurimo

19.1 New Training Methods for the HMMs

The training of the context-independent phoneme models for a minimal recognition error rate is difficult, because the variability of the phonemes in different conditions and contexts is substantial and the output densities of different phonemes do also overlap. A structure that can automatically adapt to all the complicated density functions, has a vast number of parameters and for proper estimation, the quality and quantity of the available training data is crucial. The size of the models and the training database demand robustness to the initial parameter values in order to avoid an excessively large number of training epochs and long training times.

The problem in practice with the widely spread training algorithms such as the segmental K-means (SKM) [8] and the segmental Generalized Probabilistic Descent (SGPD) [1] is that they sometimes converge slowly to low error rates unless good initial models are available.

Several common initialization methods have been compared for the mixture density hidden Markov models (MDHMM). The best results in terms of quickly obtained low final error rates in the automatic speech recognition (ASR) tests were obtained by using the Self-Organizing Maps (SOM) [2] to first train phoneme dependent codebooks and then use the codebook vectors as kernel centroids for the mixture densities. If the Learning Vector quantization (LVQ) [2] is used in the training after the SOMs, small improvements in the initialization can be achieved, but the SOM training can be performed much faster, because each phoneme codebook can be individually trained as a small SOM.

19.1.1 The Segmental SOM Training

The developed segmental SOM training for the HMMs [5] resembles to the conventional SKM type Viterbi training, but the main difference is that the parameters of mixtures belonging to the neighborhood of the best-matching component are also adapted. The motivation for the neighborhood adaptation is the parameter smoothing, where the level of the smoothing compared to the fitting accuracy to the training data is controlled by the neighborhood size. A wide neighborhood at the beginning ensures also that all the available codebook units will be drawn into useful regions in the input space. Compared to the codebooks trained without smoothing (e.g. by SKM) the accuracy provided by the best-matching Gaussian is usually worse, but that of the next $(K-1)$-best matches will be better, however, providing generalization for slightly discrepant characteristics of the test data.

The motivation to have ordered density codebooks is to enable accelerated state pdf estimation. In practice, a set of few best-matching kernels tend to dominate the density estimates for high-dimensional Gaussian mixtures, and thus the densities can be well approximated by excluding the other kernels. Since the search for the K-best matches consumes a significant part of the total computational load, the search speed-ups have a significant effect on the total recognition speed. By exploiting the similarity of the successive feature vectors and the SOM topology in the mixtures, the approximate location of the K-best candidates can be determined accelerating significantly the state pdf estimation [5]. As the radius of the applied neighborhood function decreases gradually to zero the fine structure of the topology is lost due to the folding that increases the density estimation accuracy. However, some coarse structure will still be available to maintain smoothness and search acceleration capabilities (see Figure 33).

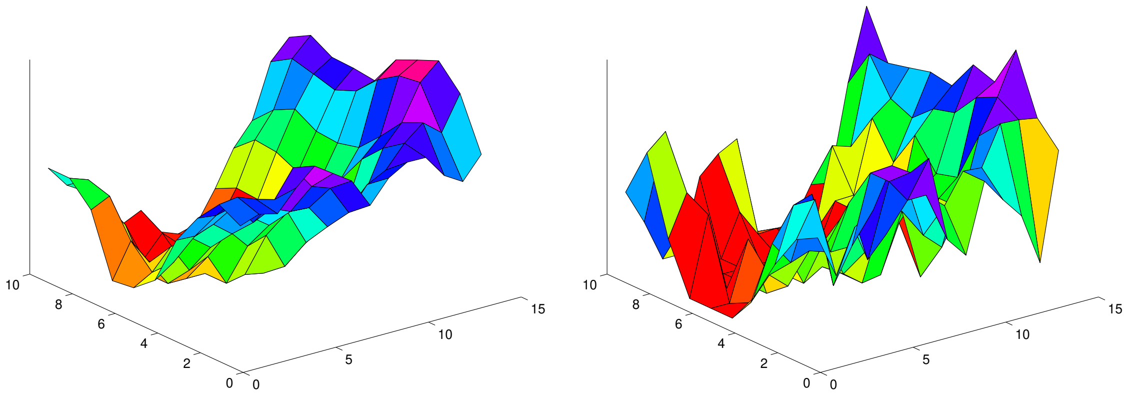

Figure 33: The responses of the individual mixture density components in the phoneme /A/ codebook organized into 10x14 grid are plotted for one randomly selected input vector. The first plot (left) is the situation when the radius is decreased to one and the second is after training with zero neighborhood

19.1.2 The Segmental LVQ3 Training and the LVQ2 Based Tuning

The segmental LVQ3 training [4]is in many ways similar to the segmental GPD improving the HMM parameters iteratively by comparing the best paths through the HMM states to the path producing the correct phoneme sequence for each training sample, updating the parameters and computing new paths. One of important difference is the lack of discrimination for situations, where the models already behave correctly in order to avoid extensive amount of adjustments to lower the state likelihoods. An other important difference is that the tuning is not directly dependent on the exact extent of the derivative of the whole word misclassification measure [1], but only on the relative difference of the modifiable parameter values to avoid the risk of improper learning step sizes for misses of variable error degree in one word.

The learning in the segmental training by both SOM and LVQ is here made in the batch mode, where each epoch includes the entire training data. The other possibility is to use a variable learning rate parameter to relate the modifications due to different training words. A proper definition of the learning rate would be difficult, however, because the parameter changes affect to the subsequent word segmentations. One method is, however, developed to train MDHMMs by LVQ that follows a pre-specified learning rate schedule between the training words. The method applies the LVQ2 type learning law to enhance the models by stochastic

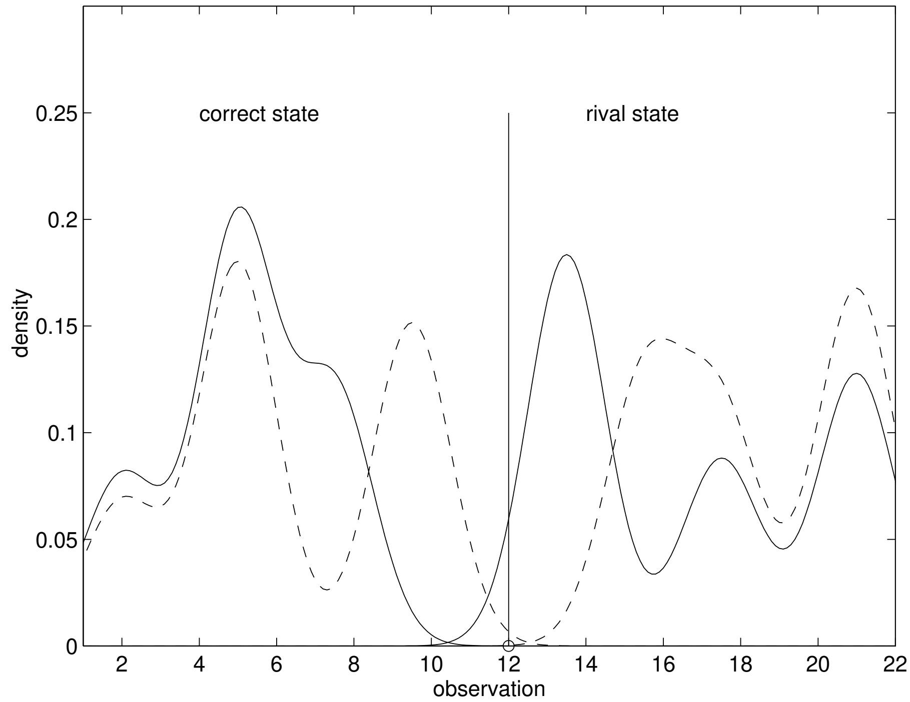

Figure 34: Adjusting the mixture densities of the competing states regarding to one observation (here, value 12). The parameters to be modified are the centroids of the nearest Gaussian for the correct state and, if a rival HMM state causes a misrecognition, also its corresponding centroid. The mixture weights of the modified mixtures are tuned respectively, but taking care of the normalization. The resulting new pdfs are shown dashed for this simple one-dimensional three-mixture case.

learning steps derived from the detected misrecognitions. This is suitable for a corrective fine tuning method, if the avoidance of the over-fitting in the training data can be controlled.
The criteria in the evaluation of the segmental training algorithms (see Part I of the Table 9) were the obtained average error rate for speakers on the both databases and that the low error rate level is achieved quickly even with initially inaccurate models. The suggested training in which the MDHMMs are first initialized by the SOMs and then trained by the segmental LVQ performed better than conventional methods with K-means initialization and SKM or SGPD training. By mixing the segmental training algorithms so that the models obtained by one is fed as an initialization to another, combinations can be found that eventually give lower error rates than the individual methods (Part II of the Table 9), but this requires much more training effort.

19.2 Increasing the Recognition Speed by Optimizing the Codebook Structure

When the dimension of the feature vectors and the size of the density codebooks are increased for better recognition accuracy, the bottle neck in online operation is the density approximation made by each HMM state for each feature vector in the

Initialization algorithm	HMM training algorithms	Error rate%	
		Data 90 basic	Data 95 context80
Part I			
KM	SKM	6.0	6.2
KM	SGPD	7.1	5.8
SOM	SLVQ3	5.6	5.3
Part II			
KM	SKM+SGPD	7.3	5.4
SOM	SLVQ3+SGPD	7.3	4.8
KM	SKM+LVQ2	5.5	5.6
SOM	SLVQ3+LVQ2	5.4	5.2

Table 9: Average test set error rates for alternative training methods after the initialization by K-means or SOM. The training methods are segmental K-means, segmental GPD, segmental LVQ3 and the corrective tuning based on LVQ2. In the Part I, the 5 epochs of HMM training is applied (no significant improvements was detected between 5 and 10 epochs). In the Part II, the last 5 training epochs were made with another algorithm (the improvement is significant, except for applying LVQ2 after the LVQ3) and the final error rates are given.

observation sequence. The topological K-best search was presented in [5] to give an example of a way to utilize the topology of an organized codebook for a fast approximative search algorithm for large codebooks. In addition of the topological order, this method assumes also that the successive feature vectors of speech usually resemble each other. Briefly, the search method presented in the Figure 35 begins by re-ranking the previous K-best matches and continues by checking the neighbors of the currently best match. If a new best is found, also the new neighbors are checked. This process continues until no more new best matches are found [6]. A complete search through the codebook is performed periodically to react for abrupt feature changes [7].

In the K-best search the fastest search time can be expected, if the candidates are ordered so that the most likely winners are checked first and the components of the feature vectors are processed in the order of decreasing significance. These characteristics are important, because each individual check of one candidate can be aborted immediately, when it becomes evident that it is not part of the K-best. With no special knowledge about the rank of the candidates except the continuous character of the signal, a good performance can be expected, if the candidates are scanned according to the distance in SOM topology from the expected winner. Similarly with no special knowledge about the rank of the components, it is best to orgarnize them according to the decreasing variance, in general.

The frequency of the complete search affects to the ability to react to fast changes in the signal characteristics and is, along with the number of the K-best matches and the size of the basic search neighborhood, a controllable variable to increase the accuracy or the speed of the search.

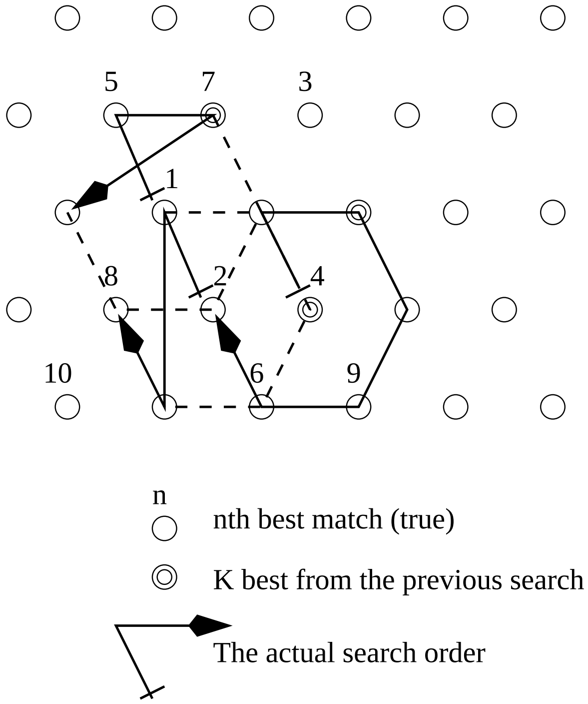

Figure 35: The topological search order for SOM codebooks.

The tree-search SOM [3] suits well to the fast approximative search for large codebooks, because the tree structure offers $O(\log N)$ search complexity instead of the normal $O(N)$. The possible loss of accuracy may follow from the sequential branching decisions by which most of the units are eliminated from the individual inspection. For K-best search the effect of the branching decisions is softened by expanding the search into the lower layer search areas associated with the rival best-matches from the upper layer. For density approximation purpose the Gaussian kernels are trained only for the lowest SOM layer and the upper layers act only as a search tool.

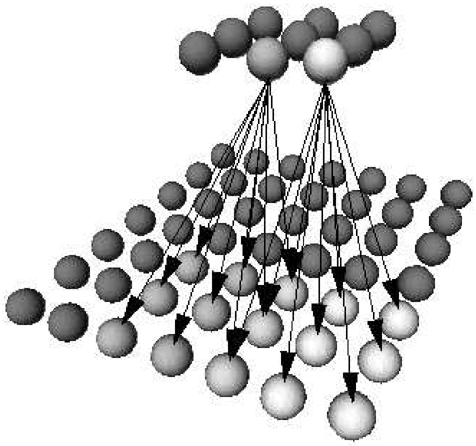

Figure 36: Two layers of the Tree-search SOM. In this map, each upper layer unit has a grid of 9 child units.

The results from the experiments indicate that the Tree-search SOM can be used as

a slightly worse performing, but a faster substitute to the normal SOM codebook. In the comparison by a ASR test to the corresponding normal SOM codebook, the Tree-search SOM, in which the recognition time decreased by 20%, increased the average number of recognition errors by 14%.
The topological K-best search compared to the unordered complete search offers a speed-up in the ASR experiments about 30–60% depending on the mixture sizes and feature vectors, while the increase of the average number of errors is only 4–10%. Despite the loss of most of the codebook topology, after the segmental LVQ3 training the same topological K-best search provide about 10% less recognition errors (about the same error rate as by complete search before the LVQ3). Thus, fortunately, the LVQ training seems to be more efficient to reduce the errors by increasing the discrimination than it is to generate them by destroying the topology required for fast search.

References

[1] W. Chou, B. Juang, and C. Lee. Segmental GPD training of HMM based speech recognizer. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages 473–476, San Francisco,USA, april 1992.

[2] T. Kohonen. *Self-Organizing Maps*. Springer, Berlin, 1995.

[3] P. Koikkalainen and E. Oja. Self-organizing hierarchical feature maps. In *Proceedings of the International Joint Conference on Neural networks (IJCNN)*, volume II, pages 279–285, Piscataway, NJ, 1990. IEEE Service Center.

[4] M. Kurimo. Segmental LVQ3 training for phoneme-wise tied mixture density HMMs. In *European Signal Processing Conference*, volume 3, pages 1599–1602, Trieste, Italy, September 1996.

[5] M. Kurimo and P. Somervuo. Using the Self-Organizing Map to speed up the probability density estimation for speech recognition with mixture density HMMs. In *Proceedings of the International Conference on Spoken Language Processing*, volume 1, pages 358–361, Philadelphia, PA, USA, October 1996.

[6] J. Lampinen and E. Oja. Fast self-organizing by the probing algorithm. In *Proceedings of the International Joint Conference on Neural networks (IJCNN)*, volume II, pages 503–507, Piscataway, NJ, 1989. IEEE Service Center.

[7] E. Lopez-Gonzalo and L. A. Hernandez-Gomez. Fast vector quantization using neural maps for CELP at 2400 bps. In *Proceedings of 3rd European Conference on Speech Communication and Technology*, volume 1, pages 55–58, Berlin, Germany, September 1993.

[8] L. Rabiner, J. Wilpon, and B. Juang. A segmental K-means training procedure for connected word recognition. *AT&T Technical Journal*, 64:21–40, 1986.