Self-Organizing Maps and Learning Vector Quantization for Feature Sequences

Panu Somervuo (panu.somervuo@hut.fi) and Teuvo Kohonen
Helsinki University of Technology, Neural Networks Research Centre, P.O. Box 2200, FIN-02015-HUT, Finland

Abstract. The Self-Organizing Map (SOM) and Learning Vector Quantization (LVQ) algorithms are constructed in this work for variable-length and warped feature sequences. The novelty is to associate an entire feature vector sequence, instead of a single feature vector, as a model with each SOM node. Dynamic time warping is used to obtain time-normalized distances between sequences with different lengths. Starting with random initialization, ordered feature sequence maps then ensue, and Learning Vector Quantization can be used to fine tune the prototype sequences for optimal class separation. The resulting SOM models, the prototype sequences, can then be used for the recognition as well as synthesis of patterns. Good results have been obtained in speaker-independent speech recognition.

Keywords: learning vector quantization, self-organizing map, sequence processing

1. Introduction

The Self-Organizing Map (SOM) consists of a regular, usually two-dimensional grid, onto which a distribution of input items is projected nonlinearly. The mapping tends to preserve the topologic-metric relations between input items.

The projection is made by a matching process. With each grid unit, a generalized model is thought to be associated. For each input item, the closest model in some metric is identified. The collection of models is optimized to approximate all inputs.

It may be well-known that SOMs can be constructed using any generalized distance function defined between the input items [4](pp. 118-121), and that the updates can be made in batches [4](pp. 127-128). It has recently been pointed out in [5, 6] that once the distance function is defined, the averages in the Batch Map method can be evaluated as generalized "medians" over batches of samples, e.g. symbol strings. The SOM of symbol strings gave the motivation for the present work. Introduction and experiments of the SOM of symbol strings have been presented in [5, 6]. For a textbook account of the SOM and the LVQ, see [4].

The idea of using generalized medians in the Batch Map type SOM algorithm comes from [5]. One batch round of the algorithm is shortly described as follows. Every input item is first listed under the corresponding best-matching unit (BMU). Then for each map unit, a new prototype is taken as the "centermost" item, e.g. the item having the smallest sum of generalized distances to other items belonging to the union of the lists of the neighboring nodes. In the definition of the centermost member of the list, the sum of squared distances, or maximum distance to other sequences can also be used.

In this work the models associated with the grid points were taken as sequences of real feature vectors. For input sequences that vary in length and rate, the closest model sequence in the SOM is found by dynamic time warping (DTW) [11]. The DTW is also used for forming averages of sequences. The resulting prototype sequences can be used as reference templates in both pattern recognition (e.g. speech recognition) and synthesis (e.g. speech production). Although time signals are here of main concern, warping can also be made in other dimensions. As pointed out in [1], many static processes can be reinterpreted as dynamic processes in which an artificial time coordinate is introduced.

The novelty here is to associate a feature vector sequence with each SOM node. In addition to the generalized median, an arithmetic average can be defined for feature vector sequences with different lengths [12]. Therefore both incremental learning and the Batch Map method can be used.

Previous studies on the clustering of DTW templates can be found in [10]. Work with the LVQ and DTW has been done in [8] using the Generalized Probabilistic Descent framework. Contrasted with the aforementioned references, the present work also takes into account the averaging of the temporal structure of sequences.

2. Feature sequences and the SOM

In its original form, the SOM was suggested for an algorithm to construct a projection of the distribution of static input vectors, and the correlations between successive input vectors were not considered. Later, especially in speech recognition, several approaches have been made to take the dynamics of the input signal into account.

Several experiments have shown that if a set of feature vectors over a time window is concatenated into a higher-dimensional pattern vector, the recognition results are improved considerably [9]. In most implementations, however, the context has been fixed by the predefined number of short time feature vector frames. Therefore, e.g. in speech recognition, the speaking rate still affects the recognition result.

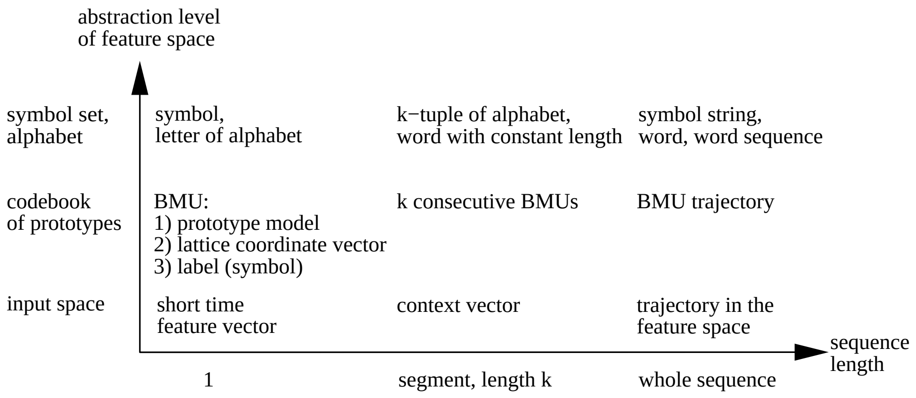

Figure 1. SOMs can be constructed for different abstraction levels of the input. This diagram illustrates what kind of model can be associated with each SOM node.

Another straightforward way to handle sequential data is to project a feature sequence onto the SOM by finding the best-matching unit (BMU) for each instantaneous feature vector. Classification of the feature sequence can then be made by means of the BMU-trajectory, e.g. the sequence of the BMU-coordinates on the map can be used as a new feature to be classified [13]. This solves the problem of speech rate. However, because the BMU-trajectory is a projection from the input feature space onto the map, some information is lost. If one BMU is replaced by several winners or by the activation image of the whole map [2, 3], more information can be preserved. This can be considered as a change of the basis in the feature space. Prototype vectors of map items now constitute new basis vectors and each feature vector is represented in this (redundant) map "base". When the activation image of the SOM units is integrated (leaky integration with an appropriate time constant), the result is a short time feature histogram which can be fed as an input to an upper layer map.

Still better recognition results can be obtained, if instead of the BMU-trajectories, the trajectories in the original feature space can be used. Similar model trajectories must then be used as models at the map units, and comparison of the trajectories must be based on, e.g., the dynamic time warping (DTW), see Sec. 3. However, now we also have to introduce a method to update the SOM models. This is one of the main ideas that are introduced in this work.

In order to gain a view of different representations, levels of abstraction, and sequences for which a SOM can be constructed, cf. Figure 1.

3. Dynamic time warping

In most real-world applications, normalization of two sequences into the same time scale ("time alignment") is based on dynamic programming (DP). The first ideas of DP have been presented in [1], but the basics can also be found in [11, 15, 12].

Dynamic time warping (DTW) is a DP-based pattern matching algorithm with a nonlinear time normalization effect [11]. In DTW, sequences are matched against reference templates. Replacing the reference templates by chains of states leads to hidden Markov models (HMMs) and the Viterbi algorithm [14]. Using the symbols instead of the feature vectors in the sequence results in the Levenshtein distance [7].

3.1. DISTANCES BETWEEN SEQUENCES

Let two feature sequences A and B consist of M and N feature vectors, respectively: $A = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_M]$ and $B = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_N]$. The distance between two sequences is computed along a warping function, which can be depicted as a path in a two-dimensional trellis. This function is computed by the dynamic-programming algorithm. After that, the time-normalized distance between sequences A and B is defined as

$$D(A,B) = \min_P \left[\frac{\sum_i \sum_j w(i,j) d(\mathbf{a}_i, \mathbf{b}_j)}{\sum_i \sum_j w(i,j)} \right], \tag{1}$$

where $w(i,j) > 0$ if trellis point (i,j) belongs to the warping path P, and $w(i,j) = 0$ otherwise. Here $d(\mathbf{a}_i, \mathbf{b}_j)$ is a distance between instantaneous feature vectors $\mathbf{a}_i$ and $\mathbf{b}_j$.

Several constraints can be added to the warping path P. Probably the most natural constraint is to require that it must be continuous. In discrete-time sequences continuity is understood as connectedness. The beginning and end points of the warping path can be fixed. The number of possible paths in variation can also be reduced by using various slope constraints. There is some indication that the utmost minimization of (1) does not necessarily give the best recognition results, and the use of additional constraints to the warping path, like slope constraints, may increase the recognition rate [11]. This is because in classification one tries to find the correct classes by matching the observations against the reference models, and the discrimination of the classes is more important than the minimization of the quantization error.

3.2. ARITHMETIC AVERAGE OF TWO SEQUENCES

The average of two sequences can be formed by computing the sequence of weighted averages of feature vectors along the warping path [12]. Let $\mathbf{a}_i$ and $\mathbf{b}_j$ be two corresponding feature vectors in the warping path. Their weighted average is defined as

$$\mathbf{c}_k = q\mathbf{a}_i + (1-q)\mathbf{b}_j, \tag{2}$$

where q is a real number between 0 and 1. Let us denote the sampling instants of $\mathbf{a}_i$ and $\mathbf{b}_j$ by t_i and t_j, respectively. The corresponding sampling instant of $\mathbf{c}_k$ is then

$$t_k = qt_i + (1-q)t_j. \tag{3}$$

Because the DTW tolerates durational differences, accurate timing is not too important if the prototype sequences are used in recognition. But if they are used in a synthesis task, timing is more important. In order to define consecutive vectors in the average sequence $C = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K]$ at regularly spaced sampling points, the average feature vector corresponding to the desired time instant can be interpolated between the nearest time instants t_k and t_{k+1} found in the discrete warping path. Linear interpolation was used in the current work. Figure 2 illustrates the averaging of two sequences.

It would be straightforward to expand the average of two sequences to the average of N sequences by using an N-dimensional DP-trellis. That would, however, require a great amount of computation for large N. The average of several sequences can be approximated by iteratively using pairwise averaging and letting the value of q be a function of the number of already averaged sequences.

4. DTW-SOM and DTW-LVQ

In the present modification of the original SOM, henceforth named the DTW-SOM, each map node does not contain a single feature vector, but a complete feature vector sequence. The dimension of the feature vector is fixed, but the length of the model sequence may vary during learning. The whole feature sequence prototype is adapted towards the input feature vector sequences. Referring to equations (1), (2) and (3), the input sequence is denoted by A, and the prototype sequence is denoted by B before adaptation and C after adaptation, respectively. The best-matching unit is the node having the smallest distance to the input sequence according to equation (1). The prototype sequences

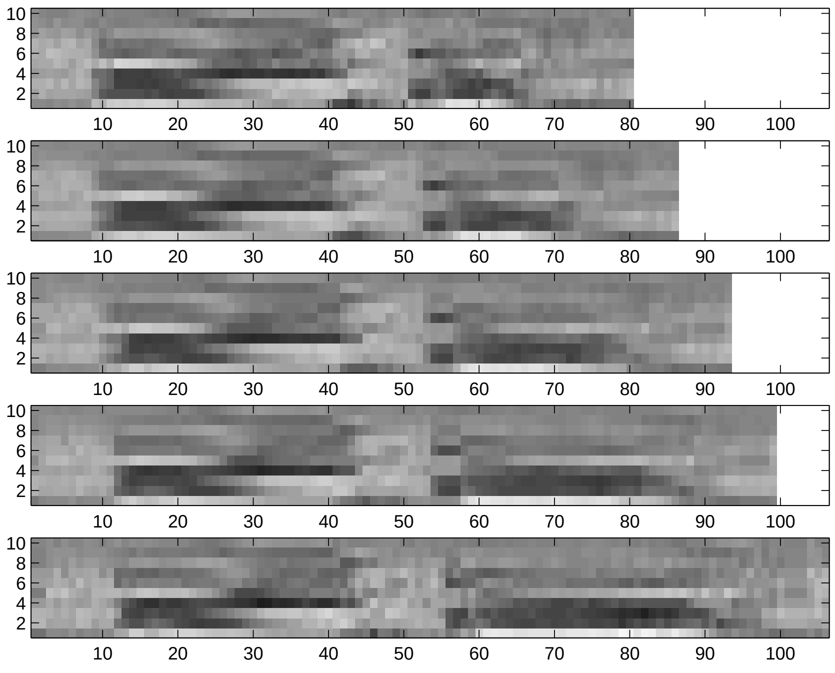

Figure 2. Averaging of two sequences. The two sequences A and B are cepstrum sequences computed from spoken Finnish words “AIKA” and “AIKAA”. They are shown on the top and on the bottom of the figure having lengths of 80 and 106 frames, respectively. Three average sequences between them are computed with $q = 0.75$, 0.50, and 0.25, respectively. Notice that the length of the average sequence is a weighted sum of the lengths of the original sequences according to the value of q. Since the beginning parts of the original sequences in this example are almost equal, these parts are preserved almost as such in the average sequence, and the time-warping is mostly concentrated to the end of the sequences.

can be adapted incrementally after each input sample sequence, or as a batch process. In the incremental learning, the arithmetic average is used, and q in equations (2) and (3) is determined by the learning rate and the neighborhood function. In the batch mode, either arithmetic averages or generalized median sequences can be used as average sequences.

The arithmetic averaging takes everything into account what is in the warping path, and every input sample sequence has an effect on the prototype sequence. One bad warping can thus collapse the whole previous prototype sequence. But if the generalized median is used as an average sequence instead, outliers have less effect. A two-stage approach is possible in which the generalized median sequence is found first and then fine tuned by incremental averaging.

The DTW-LVQ proceeds almost similarly as the DTW-SOM (although the neighborhood is not used). In the BMU search, DTW is used in the usual way, but when the average sequence is formed, the class information is taken into account. If arithmetic averaging is used, q in equation (2) is provided with the minus sign if the class label of the input sequence A does not agree with the class label of the best-matching unit, otherwise the plus sign is used. If the median sequences are used, the corresponding plus and minus signs are used when the sequence distances are summed as in [5, 6].

5. Experiments

Figure 3 illustrates a DTW-SOM obtained in the first experiment. It is a demonstration of unsupervised clustering where the temporal structure of input samples is taken into account. Input feature sequences were computed from natural Finnish speech. 10-dimensional cepstrum vectors were used as feature vectors and input data consisted of one hundred utterances of numbers from zero to nine. The SOM was initialized by 10-dimensional white noise vector sequences and it was trained by using incremental arithmetic averaging. After training the map was labeled according to majority voting. Phonetically similar words are located near each other on the SOM.

In the second experiment, speaker-independent recognition was implemented using one reference template for each word in the vocabulary. 5 female speakers and 15 male speakers had each uttered four times the 22 Finnish command words in the vocabulary. There were thus altogether 1760 utterances in the data set. Recognition tests were repeated 20 times, each time having a different speaker in the test set and the remaining 19 speakers in the training set. 10-dimensional cepstrum vectors were used as features. The purpose of this experiment was not to compare different recognition methods but instead to compare three different methods for obtaining the reference sequences once the recognition method is fixed.

Basic DTW was used with no slope constraints. The slope constraint introduced in [11] would have limited the slope of the warping path between 0.5 and 2. This implies that the length of the longer sequence may not exceed the length of the shorter sequence by a factor that is over 2, if the start and end points of the sequences are fixed in the matching. For some words in our data set the duration of the longest utterance was three times the duration of the shortest utterance.

The average recognition error of 20 independent test runs (altogether 1760 input sequences) is given in Table I. Using randomly picked

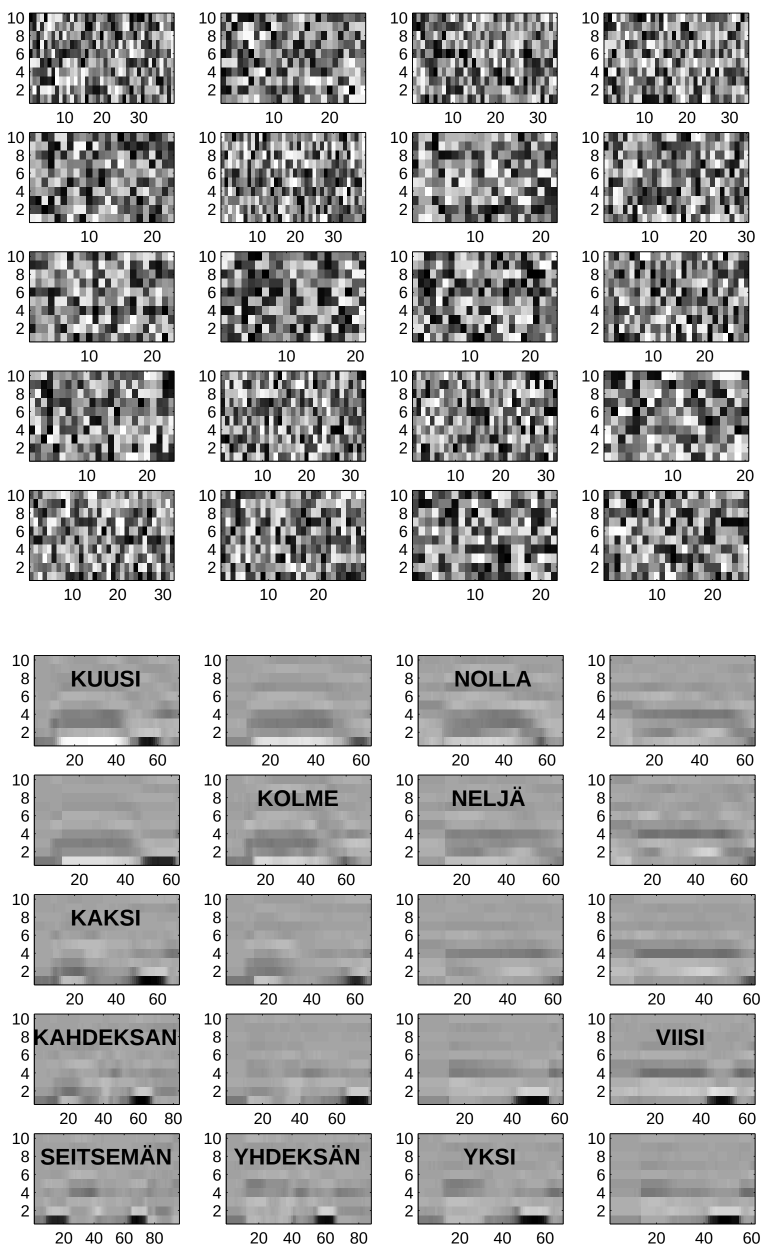

Figure 3. Prototype sequences of a DTW-SOM for 10-dimensional input feature vector sequences. The map size is 4×5 units. The initial map is shown on the top and the organized map after training is shown on the bottom. Dark shades of gray indicate low values and light shades of gray indicate high values of feature vector components, respectively. In phone /s/, the first cepstrum coefficient is a large negative number. This is shown as a black spot in the first feature vector component in the prototype sequences of the organized map.

Table I. Speaker-independent word-recognition experiment with 1760 utterances from the vocabulary of 22 Finnish command words.

Reference templates	**Error, per cent**
one randomly picked sequence from each class	18.5
one median sequence from each class	3.1
one DTW-LVQ sequence for each class	1.5

reference templates (from the correct class, however), the average recognition error for the test set was 18.5 per cent. Using median templates of the training set, the test set error was 3.1 per cent. After 3000 rounds (two times the number of the sequences in the training set) incremental DTW-LVQ fine tuning of the aforementioned median templates, the test set error was 1.5 per cent.

6. Conclusion

In this work the Self-Organizing Map and Learning Vector Quantization algorithms were constructed for data items that consisted of complete feature sequences. These sequences could have variable lengths and rates. Dynamic time warping was used to compute the distances between sequences. Due to the DTW, both durational differences in the input sequences as well as spatial variances in the feature vectors can be tolerated. The DTW-SOM can be used for unsupervised clustering of sequential input items whose temporal structure is of interest. Application areas may include speech processing, natural handwriting processing, and process monitoring. The DTW-LVQ was used in speaker-independent isolated-word recognition, where a test-set error rate of 1.5 per cent was achieved.

References

1. R. Bellman, Dynamic Programming, Princeton University Press, Princeton, New Jersey, 1957; 6th printing 1972.
2. J. Kangas, “Time-dependent self-organizing maps for speech recognition”, in T. Kohonen et al. (eds.), Artificial Neural Networks, vol. 2, pp. 1591-1594, Elsevier, Amsterdam, 1991.
3. J. Kangas, “On the analysis of pattern sequences by self-organizing maps”, PhD thesis, Helsinki University of Technology, Finland, 1994.

4. T. Kohonen, Self-Organizing Maps, Springer Series in Information Sciences, vol. 30, Springer, Heidelberg, 1995; 2nd ed. 1997.
5. T. Kohonen, "Self-organizing maps of symbol strings", Report A42, Helsinki University of Technology, Laboratory of Computer and Information Science, Espoo, Finland, 1996.
6. T. Kohonen and P. Somervuo, "Self-organizing maps of symbol strings with application to speech recognition", in Proc. of Workshop on Self-Organizing Maps (WSOM'97), pp. 2-7, Espoo, Finland, 1997.
7. V. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals", Cybernetics and Control Theory, 10, 8, pp. 707-710, 1966.
8. E. McDermott and S. Katagiri, "Prototype-based discriminative training for various speech units", in Proc. of Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'92), pp. I-417-420, San Francisco, California, 1992.
9. J. Mäntysalo, K. Torkkola, and T. Kohonen, "Mapping context dependent acoustic information into context independent form by LVQ", Speech Communication, 14, 2, pp. 119-130, 1994.
10. L. Rabiner and J. Wilpon, "Considerations in applying clustering techniques to speaker-independent word recognition", J. Acoust. Soc. Am., 66, 3, pp. 663-672, 1979.
11. H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition", IEEE Trans. Acoustics, Speech, and Signal Processing, 26, 1, pp. 43-49, 1978.
12. D. Sankoff and J. Kruskal, Time warps, string edits, and macromolecules: the theory and practice of sequence comparison, Addison-Wesley, 1983.
13. K. Torkkola and M. Kokkonen, "Using the topology-preserving properties of SOFMs in speech recognition", in Proc. of Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'91), pp. 261-264, Toronto, Canada, 1991.
14. A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm", IEEE Trans. Inform. Theory, 13, pp. 260-269, 1967.
15. G. White, "Dynamic programming, the Viterbi algorithm and low cost speech recognition", in Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'78), pp. 413-417, Tulsa, Oklahoma, 1978.