Redundant Hash Addressing of Feature Sequences using the Self-Organizing Map

PANU SOMERVUO (panu.somervuo@hut.fi)
Neural Networks Research Centre, Helsinki University of Technology, P.O.Box 2200, FIN-02015 HUT, Finland

Abstract. Kohonen's Self-Organizing Map (SOM) is combined with the Redundant Hash Addressing (RHA) principle. The SOM encodes the input feature vector sequence into the sequence of best-matching unit (BMU) indices and the RHA principle is then used to associate the BMU index sequence with the dictionary items. This provides a fast alternative for dynamic programming (DP) based methods for comparing and matching temporal sequences. Experiments include music retrieval and speech recognition. The separation of the classes can be improved by error-corrective learning. Comparisons to DP-based methods are presented.

Keywords: music retrieval, redundant hash addressing, self-organizing map, sequence processing, speech recognition

1. Introduction

Temporal sequences arise from various kinds of sources in nature. Sensory elements transform the events into measurements and corresponding feature vectors. This paper addresses the question of how to efficiently process the feature sequences. Applications include retrieval, error correction, and recognition of sequential data. Due to the durational differences in the feature sequences and the variation and noise in the feature vectors, both temporal and spatial fluctuations must be tolerated in the sequence comparison. Dynamic programming (DP) based methods provide solutions for this, but they can be computationally heavy. A different approach is to use local fixed-sized features of the sequence. This facilitates the use of fast associative methods.

The present work combines two methods developed by Teuvo Kohonen. These are Redundant Hash Addressing (RHA) and the Self-Organizing Map (SOM). RHA is an associative method based on the use of multiple features extracted from the same input item. It was introduced in 1978 as a fast method for the recognition and correction of garbled symbol strings [2]. The SOM is an artificial neural network with neurons located at the nodes of a low-dimensional lattice. Each node contains a reference model representing a local domain of input space. The SOM algorithm was introduced in 1981 [3].

RHA has mainly been used for correcting textual output from speech recognizers, see e.g. [4]. In these, the recognition result is already in the form of a phoneme string. But in some applications, e.g. music processing, it is more difficult to extract, or even define, the underlying symbol sequence. Nevertheless, we shall demonstrate now that the RHA principle can still be used. As the RHA makes use of the N-grams[1] of symbols and therefore the feature vectors must first be quantized, the SOM is used as a codebook to map the input feature vectors into the finite set of prototype vectors. When each SOM node is provided with an index, feature vector sequences can be mapped into symbolic index sequences. Each feature vector is encoded by the index of its best-matching unit (BMU). The node indices of the SOM are thus the alphabet of the system.

RHA was first introduced for processing symbol sequences [2]. If feature vectors are used instead of the symbols, the result is then somewhat similar to the Cerebellar Model Articulation Controller (CMAC) introduced by Albus [1]. Here, however, the SOM is used to quantize and encode the input features. Error-corrective learning of association weights is presented for improving the class separation. Comparisons to the dynamic programming based methods are also presented.

2. The Self-Organizing Map

The SOM was introduced by Kohonen in 1981 [3]. It has the roots in biological nervous systems. The SOM performs a nonlinear mapping of high-dimensional input data domains onto the nodes of a low-dimensional map lattice. Each lattice node contains a reference model representing a local domain of input space. The models become ordered and specifically tuned to input signal patterns through an unsupervised learning process. A textbook [5] discusses the SOM in detail. Only some implementational features of the algorithm are described here.

Each SOM node contains the prototype vector of the feature space, the coordinate vector of the map lattice, and the label of the node. The best-matching unit (BMU) is computed by using the prototype vector in the feature space. The lattice coordinate vector is used when computing the neighborhood on the map. If the SOM lattice is structured, it is possible to directly determine the neighborhood of the nodes without performing Euclidean distance computations between lattice coordinates. This is, however, just a matter of the implementation of the algorithm. Instead of a single symbol as a label, a soft labeling can be

[1] An N-gram is an N-tuple of the alphabet, i.e. a string with length of N symbols.

used, in which case each map node contains a vector with probabilities to all symbol classes [7].

The batch version of the SOM [5] was used in the experiments. One batch round consists of two steps. First it is determined to which map node's Voronoi region each input sample belongs. This is done for all input samples before an updated prototype vector is computed as an average of the input samples belonging to the neighborhood of the map node. An efficient implementation for this is to first compute the average vector in each Voronoi region separately and then compute the final average over the union of Voronoi regions as a weighted average of the foregoing averages. This is expressed as

$$\boldsymbol{m}_c = \frac{\sum_i n_i h_{c,i}(t) \boldsymbol{m}_{i,0}}{\sum_i n_i h_{c,i}(t)}, \tag{1}$$

where n_i is the number of input samples in the Voronoi region of map node i, $h_{c,i}(t)$ is the neighborhood function, and $\boldsymbol{m}_{i,0}$ is the average vector in the Voronoi region of map node i (0 denotes the average with zero neighborhood).

3. Redundant Hash Addressing

The RHA principle proposed by Kohonen was introduced in 1978 [2, 5]. The central idea in the method is to extract multiple features from the same input item. The comparison of the input item against the reference items is based on these features. In case of character strings, segments of N consecutive letters (N-grams) have been used. The RHA system consists of the N-gram table and the dictionary. Figure 1 illustrates the method. Multiple features (N-grams) are extracted from the input string and each extracted N-gram associates the input string with the dictionary items.

A distance measure related to the RHA is called the *feature distance* (FD) [2]. It is defined between the sequences A and B as

$$FD(A, B) = \max(n_A, n_B) - n_e \ , \tag{2}$$

where n_A and n_B are the numbers of N-grams in the sequences A and B, respectively, and n_e is the number of matching N-grams. The winning dictionary item is chosen by first summing the associations activated by the input string in order to get n_e and then finding the item with the smallest distance according to Equation (2).

The RHA method is very fast, because it does not compare the input string directly against all reference strings in the dictionary.

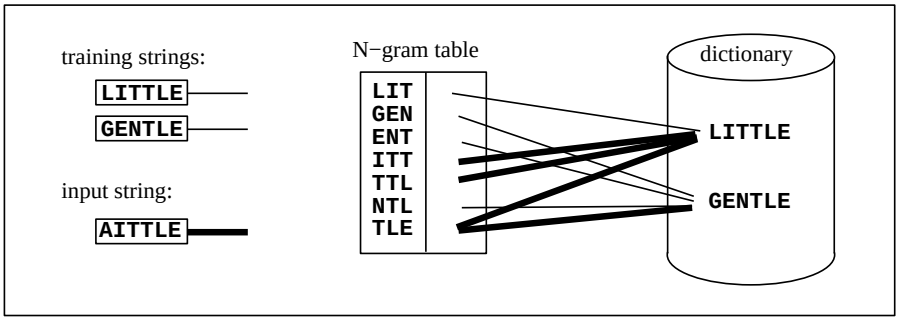

Figure 1. The RHA principle applied to character strings. The N-gram table is constructed by extracting the N-grams from training strings. Here trigrams are used ($N = 3$). From each item in the N-gram table there are associations (pointers) to the dictionary items. Associations activated by the erroneous input string 'AITTLE' are depicted by thick lines.

Comparison can be restricted to only those strings which share common features. The redundancy of the extracted N-grams enables the error tolerance. It suffices that only some of the N-grams in the input string remain free from defects.

The target of the association may be freely chosen. The dictionary can be a sequence storage or contain only the classes of the recognition task. Other associations are also possible.

4. The SOM-RHA system

The proposed system consists of the SOM, BMU N-gram table, and the dictionary. The SOM encodes the input feature vector sequence into the BMU index sequence. The BMU N-gram table contains pointers to the dictionary and the N-grams extracted from the BMU index sequence associate the input sequence with the dictionary items.

If N is 1, there are direct association pointers from the nodes of the SOM to the dictionary. In order to increase the generalization capability of the system, instead of using only one BMU for each feature vector, several map units can be used. The associations can then be weighted as a function of the quantization error of the input feature vector. But if N is greater than 1, it is more difficult to smoothen the response of one N-gram item among several N-gram items. In this work only one N-gram item is used for each segment of N consecutive feature vectors. This consists of the BMUs of the feature vectors.

4.1. IMPLEMENTATION OF THE N-GRAM TABLE

The N-gram table is constructed by extracting all N-grams from the training sequences. All extracted N-grams from one training sequence have the same dictionary item as a target of association. The associations are implemented as pointers from N-gram items to the dictionary items. Each N-gram has a position in the input sequence and therefore a position array is attached to the N-gram item. The dictionary item pointers are stored as lists in the position arrays of the N-gram items corresponding to the locations of the extracted N-grams in the training sequences. The data structure is shown in Figure 2.

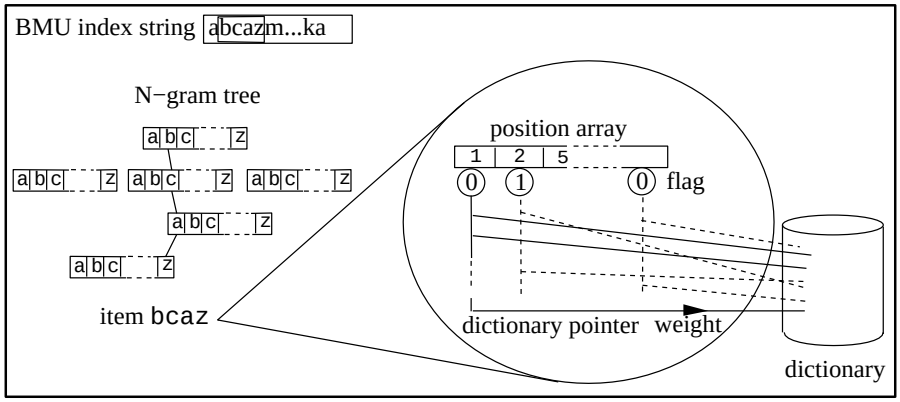

Figure 2. The N-gram table as a tree structure. N-gram items are at the leaves of the tree. Here N is 4. One N-gram item is shown enlargened. In sequence recognition, all N-grams are extracted from the BMU index string which associate the input sequence with the dictionary items. Each dictionary item contains a field where to sum the weights of the activated associations.

The N-gram table can be stored in the computer memory in several ways. An arithmetic hashing function was used in [2] to get the memory location of each N-gram item. Here the N-gram table is constructed as a tree structure, which replaces the use of the arithmetic hashing function by the tree search when locating the memory address of the BMU N-gram item. Collisions due to the hashing function are completely avoided. The tree structure allows also a flexible way to simultaneously use several N-grams with different values for N. The height of the tree corresponds to the value of N.

Searching the memory location of one N-gram item, each symbol position in the N-gram corresponds to one level in the tree. One tree node contains an array of pointers to the next level nodes. Each array element corresponds to one symbol. If the array contains all symbols of the alphabet, the next level node can be accessed directly by the value of the symbol in the N-gram because the symbols are now map unit

indices from 0 to the number of units in the map minus 1. No search along the array is then needed. The memory location of one N-gram item can thus be found by using only one direct array index at each tree level.

Keeping the position array of the N-gram item sorted enables a fast access to the dictionary item pointers in the desired position range because binary search can then be used. Another possibility is to also use tree search here.

4.2. HISTOGRAM VERSUS TRAJECTORY

In [2, 9], two N-grams were considered to match only if their positions in the strings differed no more than d positions, where d was a predefined displacement tolerance. It does not have to be a constant, it can be, e.g., a function of the position. The larger the value of d, the more tolerant the matching is to durational differences. If the value of d exceeds the length of the sequences, the comparison of the trajectories reduces to the comparison of histograms. It can be argued that the selection of d requires some a priori information about the durational differences of the sequences to be matched.

Instead of treating d as an absolute position difference, it can be used as a relative position difference tolerance computed from the previous match. Following the ideas in [11, 12], where the dictionary was built by using an N-gram network, d is then interpreted as the radius of the search neighborhood in the N-gram network.

4.3. PRESERVING THE TRUE DISTANCE

The feature distance (2) has been proved to be a mathematical distance function [9]. In the proof, however, it was required that no N-gram occurs twice in the same string. Later in the same paper it was though noted that this requirement can be relaxed if the positions of the matching N-grams are unique. Duplicate N-grams may be rare in phoneme strings, but using long unsegmented (uncompressed) feature sequences they are more likely to occur. In speech recognition, for example, during the steady part of speech like the middle of a vowel, one particular SOM node may be the BMU for several consecutive feature vectors. In order to satisfy the requirement for the true distance, and more importantly, in order to improve the performance of the system, it must be ensured that the match of one N-gram item is counted only once. This was solved here by using a flag attached to each element of the N-gram item position array telling whether the match has occurred or not, see the data structure in Figure 2. It is important to note that this flag needs not to be attached to each association pointer starting from the

N-gram item. That would slow down the performance because the flags must be initialized before each sequence recognition. There are usually considerable more association pointers than position array elements in the N-gram table.

4.4. ERROR-CORRECTIVE LEARNING OF ASSOCIATION WEIGHTS

In the recognition task, the targets of the associations, i.e. the dictionary items, are classes. When each association is provided with a weight, the separation of the classes can be improved by adjusting the association weights appropriately. The similarity between two sequences is measured by means of discrete N-gram blocks. The idea in the error-corrective weight tuning is to decrease the overlap of the classes so that the effect of common N-grams shared by different classes is decreased and the the effect of distinguishing N-grams is increased. The only information used in the training is whether the classification (association) is correct or incorrect. No target values of feature vectors or sequences are used. The weight of the association from N-gram item i to the dictionary item j is denoted by w_{ij}. The learning rule [6] is

$$w_{ij}(t+1) = \begin{cases} w_{ij}(t) + \Delta W & \text{if class } j \text{ is correct} \\ \max(w_{ij}(t) - \Delta W, 0) & \text{if class } j \text{ is incorrect, but it is the winner} \end{cases} \quad (3)$$

ΔW is the resolution of the association weight, value 1.0 was used in the experiments. Training proceeds by computing the matching N-gram items from each input sequence and, if the classification of the whole sequence is incorrect, using Equation (3). Restricting the weights to be always positive, $\max(w_{ij}(t)-\Delta W, 0)$ in Equation (3), gave slightly better results in the speech recognition tests.

5. Experiments

5.1. MUSIC RETRIEVAL

Pieces of acoustic piano music from Johann Sebastian Bach's Well-Tempered Clavier I were used in the music retrieval experiments. 10 seconds of music was sampled from the beginning of each first 24 pieces using a 16 kHz sampling rate. Mel-scaled spectrum vectors were computed from the audio samples using 16 ms time windows with 8 ms time intervals. A 10-by-10-unit SOM was trained and the RHA tree

was constructed by using the 24 feature sequences. The test material consisted of 24 subsequences with the duration of one second. These subsequences were extracted from the previous 10-second sequences. The beginnings of the extraction were randomly chosen. The RHA search was performed using only the histograms of the BMU N-grams. For N=1, there were 2 erroneous retrievals, 4 completely correct retrievals, and 18 retrievals where the correct music piece was present but it shared a tie with incorrect retrievals. For N=2, all retrievals were completely correct. Also for the values N=3, N=4, and N=5, respectively, all retrievals were completely correct.

5.2. SPEECH RECOGNITION

The speech material used in the experiments consisted of 1760 utterances collected from 20 speakers (5 female speakers and 15 male speakers). The vocabulary was 22 Finnish words of command. The size of the vocabulary, however, does not alone provide information about the difficulty of the recognition task. There may be mutually similar words which are more sensitive to local errors than other words. Such words were present in the current vocabulary. The speech contained natural variation due to different voices and speaking styles. Some words were not pronounced carefully and there were missing word endings. The amount of silence varied in the utterances before and after the word. The biggest problem in the multi-speaker recognition task, however, was the variation in the speaking rates. For some words, the duration of the longest utterance was three times the duration of the shortest utterance. This dataset was therefore a challenge to the RHA-based recognition. The average duration of the utterances was 740 ms and the longest utterance was 1.6 s.

The speech material was divided into four sets. All experiments were repeated four times. Each time a different combination of three speech sets was used in the training and the remaining speech set was used in the testing. Five recognition tests were first performed using dynamic programming based methods for the comparison to the RHA method. This provided also information about the effect of different representations of the feature vectors. Dynamic time warping (DTW), see e.g. [10], was used in first four tests. In the first test, a 10-dimensional mel-cepstrum vector was used as a feature vector. Features were extracted at 8 ms time intervals using 16 ms time windows. In the second test, the previous feature vector was quantized by using the prototype vector of its BMU on the SOM. In the third test, the lattice coordinate of the BMU was used as a feature. In the fourth test, SOM nodes were first labeled by using segmented and phonetically labeled training sequences

and the phoneme probability vector of the BMU was then used as a feature. In the fifth test, the feature vector sequence was first transformed into the BMU index sequence and the DP-based Levenshtein distance [8] was used in the sequence comparison. This is the best baseline test for the RHA method because both methods have the same representation for the feature vectors. In the first four tests, only one reference sequence was used for each class (the centermost sequence of the class having the smallest sum of distances to other sequences of the class), but in the fifth test all training sequences were used as reference sequences. This was for the better comparison to the RHA method, because RHA allows the storing of several sequences without distinctly slowing the performance unlike the DP-based methods.

The sixth test was the RHA test beginning by training a SOM using the feature vectors extracted from the training sequences. This followed the construction of the RHA tree and the tuning of the association weights. In the experiments, both N and d were fixed. For small values of d, no training is necessarily required in order to perfectly recall the training sequences. This holds also for large values for N. But in both cases the generalization capability can be poor. Therefore d was set to 150 ms in the experiments. Weight tuning was found to considerably improve the classification performance, the experiments are shown in Figure 3 for N=1,2,3, and 4. One weight tuning epoch included all training sequences. It can be seen that the recognition error decrease dramatically after the first weight tuning epoch.

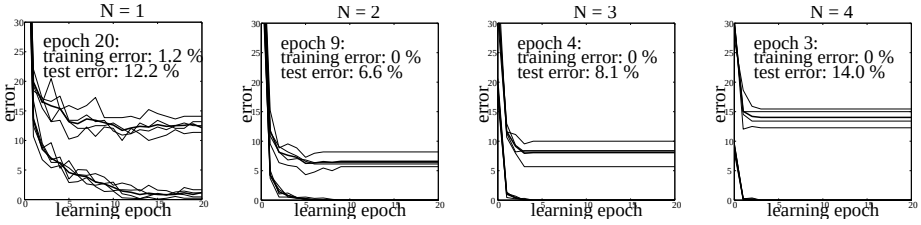

Figure 3. Weight tuning experiments for N=1,2,3, and 4, respectively. The size of the SOM was 5×5 units. Thin lines correspond to individual experiments and thick line is the average error plot. The error plots are shown for both training and test sets.

The recognition results of the six tests are shown in Table I. It can be seen from the results that increasing the size of the SOM lattice improves the recognition accuracy in tests 2-4, but decreases the accuracy in tests 5 and 6. Increasing the size of the SOM lattice decreases the size of the Voronoi regions of the map units and therefore also the quantization error. The accuracy in the tests 2-4 improves because the feature vectors are continuous valued. But in the tests 5 and 6 the features are symbolic and decreasing the size of the Voronoi regions of

Table I. Multi-speaker speech recognition experiment. Averaged results of four independent runs. Time is the average recognition time for one sequence using a 5×5 SOM. The reference sequences in test 1 were longer on the average than in test 2 which explains the time difference between these two tests.

feature, recognition method	error, per cent		time/ms
one reference sequence per class:			
1. 10-dim unquantized, DTW	3.4		78
	5×5 SOM	10×5 SOM	
2. 10-dim BMU prototype, DTW	5.3	4.5	76
3. 2-dim BMU lattice coordinate, DTW	6.6	5.0	55
4. 19-dim BMU psymbol, DTW	4.8	3.2	98
multiple reference sequences (60) per class:			
5. BMU index, Levenshtein	3.4	5.1	1060
6. RHA N=2, d=150ms	6.6	7.7	5

the map units decreases the tolerance to the variations in the speech. The RHA recognition accuracy was also experimented when d was ∞ meaning that only histograms of the BMU N-grams were used, but then the error was almost doubled.

Although the recognition accuracy of the RHA method was not as high as the accuracy using unquantized feature vectors in the test 1, the recognition time was over a magnitude smaller. It is interesting to notice that the accuracy in the test 5 using the smaller SOM was equal to the accuracy in the first test. Because the representations of the feature vectors in the tests 5 and 6 are equal, it is possible to get improvements to the results of the RHA method. It should be noted that only a single-valued N was used in the experiments and also d was fixed. As mentioned earlier, the tree structure of the N-gram table allows the use of multiple values for N simultaneously.

Tolerance to durational differences can be increased by storing the dictionary item pointers of one sequence to multiple position array elements in each N-gram item corresponding to different delays. Another possibility is to segment the feature sequence. When similar consecutive feature vectors are segmented together, each segment can be represented by one prototype vector. While the spatial resolution and robustness can be controlled by the number of nodes in the SOM lattice, the temporal resolution can be controlled by the amount of segmentation. If the segmentation is carried to extremes, each segment

is one phone or phone pair. Transforming the feature segments into phonemes leads then back to earlier works [2, 9, 4], where phoneme symbols were used as the alphabet of the RHA system.

6. Conclusion

In this work, the Self-Organizing Map was combined with the Redundant Hash Addressing principle. This contributed the main novelty of the work. The SOM encodes the input feature vector sequence into the BMU index sequence and the RHA principle is used to associate the BMU index sequence with the dictionary items. Music retrieval and speech recognition experiments were carried out as a demonstration of the method. In the speech recognition, comparisons to the dynamic programming based methods were made with success, the speed of the RHA method being its main advantage. Error-corrective tuning of association weights was found to improve the recognition results considerably.

References

1. J. Albus, "A new approach to manipulator control: the cerebellar model articulation controller (CMAC)", Journal of Dynamic Systems, Measurement, and Control, Transactions of the ASME Series G, Vol. 97, no. 3, pp. 220-228, 1975.
2. T. Kohonen and E. Reuhkala, "A very fast associative method for the recognition and correction of misspelt words, based on redundant hash addressing", in Proceedings of the 4th International Joint Conference on Pattern Recognition (IJCPR), pp.807-809, Kyoto, Japan, Nov. 7-10, 1978.
3. T. Kohonen, "Automatic formation of topological maps of patterns in a self-organizing system", in Proceedings of the Second Scandinavian Conference on Image Analysis (SCIA), pp. 214-220. Helsinki, Finland, 1981.
4. T. Kohonen, H. Riittinen, E. Reuhkala, and S. Haltsonen S, "On-line recognition of spoken words from a large vocabulary", in INFORMATION SCIENCES, Vol. 33, pp. 3-30, 1984.
5. T. Kohonen, Self-Organizing Maps, Springer Series in Information Sciences, Vol. 30. Springer, Heidelberg, 1995. Second ed. 1997.
6. T. Kohonen, Private communication, 1998.
7. J. Laaksonen, "A new reliability-based phoneme segmentation method for the neural phonetic typewriter", in Proceedings of the 2nd European Conference on Speech Communication and Technology (Eurospeech), pp. I-97-100, Genova, Italy, Sep. 24-26, 1991.
8. V. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals", in Cybernetics and Control Theory, Vol. 10, no. 8, pp. 707-710, 1966.
9. E. Reuhkala, "Recognition of strings of discrete symbols with special application to isolated word recognition", PhD thesis, Helsinki University of Technology, Department of Technical Physics, Finland, 1983.

10. H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition", in IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-26, no. 1, pp. 43-49, Feb. 1978.
11. O. Ventä, "A fast text reconstruction method for the correction of imperfect text", in Proceedings of the First Conference of Artificial Intelligence Applications (CAIA), pp. 446-452, Denver, Colorado, Dec. 5-7, 1984.
12. O. Ventä, "Associative and syntactic text correction methods for continuous speech recognition", PhD thesis, Helsinki University of Technology, Department of Computer Science, Finland, 1990.