INTEGRATING POSTERIOR FEATURES AND SELF-ORGANIZING MAPS FOR ISOLATED WORD RECOGNITION WITHOUT DYNAMIC PROGRAMMING

Serena Soldo Mathew Magimai.-Doss

Idiap-RR-17-2012

JUNE 2012

Integrating Posterior Features and Self-Organizing Maps for Isolated Word Recognition without Dynamic Programming

Serena Soldo[1,2] *and Mathew Magimai.-Doss*[1]

[1]Idiap Research Institute, Martigny, Switzerland
[2]École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
{serena.soldo, mathew}@idiap.ch

Abstract

In recent works, the use of phone class-conditional posterior probabilities (*posterior features*) directly as features provided successful results in template-based ASR systems. Moreover, it has been shown that these features tend to be sparse and orthogonal. Given such properties, new types of ASR may be investigated. In this work, we investigate the use of Self-Organizing Maps to transform sequences of posterior feature vectors representing words into sparse fixed-size patterns. We evaluate the ability of these patterns to discriminate between words in the context of template-based ASR using a simple histogram matching technique (i.e. without the use of dynamic programming). We present experiments on 75-word speaker- and task-independent isolated word recognition task.

Index Terms: Self-Organizing Maps, posterior features, isolated word recognition, fixed-size word patterns.

1. Introduction

In the context of template-based ASR, it has been shown that the use of phone class-conditional probabilities (*posterior features*), at the cost of training an estimator, can yield significantly better performance than standard acoustic features using a fewer number of templates [1]. Furthermore, it has been shown that posterior features have interesting orthogonality and sparsity properties [2]. These "binary-like" properties of posterior features could be exploited to build new type of ASR systems that go beyond the strict sequential comparison (like Hidden Markov Models or template matching based on Dynamic Time Warping, DTW). In particular, it would be interesting to use associative memory techniques together with posterior features to understand if such methods can benefit from the properties of these features.

Associative memory [3] is a recurrent neural network that can store patterns; once a pattern has been stored, it can be recalled by giving as input to the network a test pattern that is a variation or just a fraction of the stored one. However, this method in general requires that the input is sparse, binary and of a fixed length. When associative memory is used to store words (i.e. a sequence of feature vectors), we need to deal with the fact that different words have different lengths. Thus, there is a need to introduce a technique to transform the sequence of feature vectors representing a word into a pattern that satisfies the requirements for the associative memory input.

In previous works, Self-Organizing Map (SOM) [4] has been used to produce binary fixed-size patterns representing words [5, 6, 7]. More specifically, SOM has been used to project the input sequence of acoustic feature vectors onto a two-dimensional binary pattern. The training algorithm for these maps preserves topological information of the input space that may be useful in discriminating between words. The patterns obtained have been, then, classified using a Multilayer Perceptron (MLP) [5, 6] or Support Vector Machine (SVM) [7].

In this work, we perform a preliminary study to investigate the use of SOM to transform sequences of posterior feature vectors corresponding to words into fixed-size patterns (Sections 2). As a first step, we evaluate these patterns by performing a simple histogram-based word classification, where no dynamic programming is involved (Section 3). More specifically, we perform template-based isolated word recognition, where templates are represented by such fixed-size patterns instead of sequences of feature vectors (as in standard DTW-based approach). On 75-word speaker- and task-independent isolated word recognition task, this method yields up to 95.2% accuracy (Section 4).

2. System Layout

The present work investigates the use of SOM based on posterior features to model word patterns in the context of isolated word recognition. The framework is shown in Figure 1. The input sequence of feature vectors (either cepstral-based features or posterior features) is given as input to a SOM (Section 2.2) and the map of the Best Matching Units (BMU) is stored in a matrix of the same size as the feature map (Section 3).

We aim at understanding if the discrimination between words represented using these patterns can benefit from the properties of posterior features. In other words, we want to understand if the pattern obtained using posterior features can be more easily discriminated compared to the use of standard cepstral features. In order to do this we perform, for both kind of features, template-based isolated word recognition where the templates are represented using such fixed-size patterns. In the decision making step, each test pattern is compared to (previously stored) templates and the word corresponding to the "closer" template is provided as final output of the recognizer.

In the following sections we describe each element of this framework.

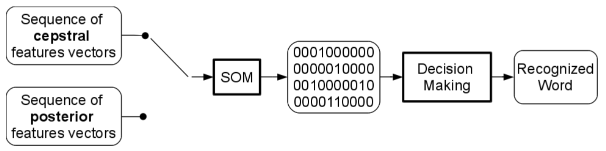

Sequence of **cepstral** features vectors
Sequence of **posterior** features vectors
SOM
0001000000
0000010000
0010000010
0000110000
Decision Making
Recognized Word

Figure 1: *Block diagram of the proposed system. A sequence of feature vectors is first transformed by SOM into a fixed-size pattern. The pattern is then compared to (previously stored) templates and the "closer" word is provided as output.*

2.1. Posterior features

Traditional cepstral-based features, like Mel-frequency cepstral coefficients (MFCCs) or Perceptual Linear Predictive (PLP) coefficients, are generally susceptible to undesirable variability, such as speaker or environment. Recently, the use of posterior features estimated using MLP directly as features was proposed [1]. They typically represent the phone class-conditional posterior probabilities of each speech frame. The standard cepstral-based feature vectors are thus transformed into linguistically meaningful dimensions.

Formally, given a cepstral-based feature vector, $x = [x^1, \ldots, x^S]^T$, and given a set of possible phone classes c_k with $k \in \{1, 2, ..., K\}$, the vector y of the posterior probabilities is given by $y = [P(c_1|x), \ldots, P(c_K|x)]^T = [y^1, \ldots, y^K]^T$. As discrete distribution, the vector y has two properties: *a)* $\forall k \in \{1, 2, ..., K\}, y^k \in [0, 1]$ and *b)* $\sum_{k=1}^{K} y^k = 1$.

In our previous work [1], we showed that posterior features can yield better performance than standard acoustic features. In particular, the use of distance measures that take into account the probabilistic nature of posterior features (such as Kullback-Leibler divergence or dot product) provides significant improvement. Furthermore, it was shown that posterior features could be efficiently estimated using MLP trained on auxiliary corpus.

2.2. Self-Organizing maps

SOM is a biologically-inspired neural network introduced by Kohonen in [8]. It aims to find a mapping from the input space to a D-dimensional map (the *feature map*) such that, after training, input vectors that are "similar" will correspond to topologically close nodes in the feature map. The mapping is defined finding the Best Matching Unit (BMU), i.e. the node (or unit) of the map that is the "most similar" to the input vector given a similarity measure.

Formally, the network consists of M processing nodes arranged in a D-dimensional map, with D typically equal to 2. Each processing node receives input from N input nodes that constitute the N-dimensional input vector $z = [z^1, ...z^N]^T$. The processing node m in the map are specified by the codebook vector $w_m = [w_m^1, ..., w_m^N]^T$ (an N-dimensional vector of weights) and their coordinates in the map lattice.

The training process follows an iterative procedure where, at each iteration t, an input vector z_i is selected from the training dataset. Usually, the winner node j in the feature map is selected as:

$$j = \arg\min_m \{d(z_i, w_m)\}.$$

where $d()$ denotes the Euclidean distance. The weights of the network are then updated according to the following rule:

$$w_m(t+1) = w_m(t) + \alpha(t)H(w_j, w_m, t)[z_i - w_m(t)]$$

where $\alpha(t)$ is the learning rate and H is the *neighbourhood function*, typically a Gaussian function which specifies the set of neighbours for each node.

Both cepstral feature vectors, x, or posterior feature vectors, y, can be used as input vector z in the SOM training (thus having $N = S$ or $N = K$, respectively). Traditionally, Euclidean distance is well suited for cepstral feature vectors. However, when posterior features are used, other distance measures that take into account the probabilistic nature of these features are better suited than Euclidean distance. Also, posterior features tend to be sparse and orthogonal [2]. In addition to it, in [2] it was also shown that dot product is the "optimal" estimation of the probability that a pair of posterior features vectors belong to the same class when MLP is used as estimator.

Motivated by these observations, in this work we investigate the use of *dot-product SOM* [4] for posterior features. In this formulation, the winner node is selected as,

$$j = \arg\max_m \{z_i^T \cdot w_m)\}$$

and the weights of the network are updated according to:

$$w_m(t+1) = \frac{w_m(t) + \alpha(t)H(w_j, w_m, t)z_i(t)}{||w_m(t) + \alpha(t)H(w_j, w_m, t)z_i(t)||}$$

where $||\cdot||$ denotes the Euclidean norm of the vector.

3. Word-map Modelling

In this section we describe a method to produce and classify fixed-size patterns representing words through the use of SOM and a simple classification framework.

3.1. Word-map generation

When a sequence of feature vectors is given as input, SOM responds activating one node of the feature map for each vector. A matrix of the same size as the feature map can be used to keep trace of these outputs. At the end of this process, the matrix will contain only 0/1 values: there will be '1' in the positions corresponding to the BMUs of each vector of the sequence. This matrix, representing a sequence of speech vectors (corresponding to a word), is a binary, sparse and fixed-size pattern.

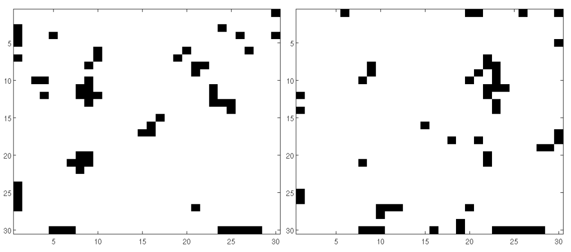
Figures 2 and 3 show examples of word-maps for two different words obtained using PLP cepstral features and posterior features, respectively. It can be noticed that when posterior features are used the variability in the maps is lower and the sparseness is higher compared to the use of cepstral features. Moreover, in case of posterior features each area of the map can be considered as locally modelling one phone class.

3.2. Word-map histogram generation and classification

Word-map histogram generation. In this work, we represent each word-map by a histogram instead of a binary map. The histogram is estimated by collecting in the matrix the count of how many times each node has been a BMU for the input sequence. Then, each value in the matrix is normalised by the number of frames in the sequence. Finally, cascading the columns of the matrix we obtain the histogram. It can be observed that such a histogram tries to retain somewhat duration information. In the reminder of this paper, when we refer to a word-map we actually refer to the corresponding histogram.

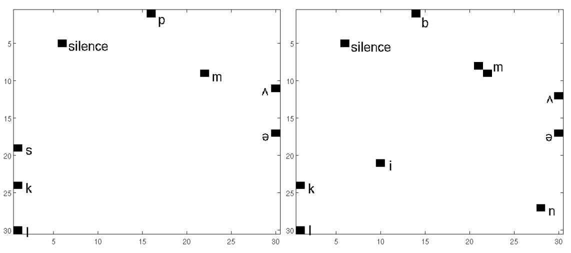
Word-map classification. The goal of this study is to investigate the discriminative power of the word-map obtained using posterior features compared to the use of cepstral features. For this reason, we prefer to adopt a simple classification framework rather than training a sophisticated classifier.

In our system, the templates and the test words are transformed into fixed-size pattern, thus we can classify each test sample simply searching the "closer" word among a set of stored templates. More precisely, we first store the word-maps of each template sample. Then, each test sample is transformed into a word-map and compared with all the stored templates. The word corresponding to the closer templates (with respect of a similarity measure) is provided as output. In standard template-based approach, words are generally represented as sequences of feature vectors and DTW algorithm is used to compute the distance between two samples (i.e. the test sample and each template). In SOM-based approach, since the word-maps are represented as histograms (with fixed length), the similarity



(a) Columbus (b) Columbian

Figure 2: *Word-maps for a sample of the word Columbus (kəlʌmbəs) and the word Columbian (kəlʌmbiʌn) using PLP features with 30×30 SOM. The number of frames for Columbus is 124. The number of frames for Columbian is 122.*



(a) Columbus (b) Columbian

Figure 3: *Word-maps for a sample of the word Columbus (kəlʌmbəs) and the word Columbian (kəlʌmbiʌn) using Posterior features with 30×30 SOM. The number of frames for Columbus is 124. The number of frames for Columbian is 122. Each label indicates the phone associated to the component with the highest value in the corresponding codebook vector of the feature map.*

measure is defined as the Bhattacharyya distance, thus no dynamic programming is involved.

Exploiting the fact that the word-maps have the same size, we investigated the possibility of storing one average template per word rather than the individual templates. In other words, given the set of templates, we compute a single average word-map for each word. To some extent, this correspond to train a model for each word, thus providing more robustness compared to the use of the individual templates. The results showed that the use of average templates always yielded the best performance. Also, since only one template per word is stored, this method requires less storage and computational time for the classification task. In Section 4.3, we only present the results using the average template for each word.

3.3. Temporal information

It can be noted that in the word-maps showed in Figures 2 and 3 the temporal information is lost (the order of activation of the nodes or how many times a node has been activated). However, such information can help to discriminate, for example, sequences with the same phones in different order.

As proposed in [5], some temporal information can be included by splitting the input sequence into two parts and mapping each part into a different matrix. In this case, the sequence of feature vectors is first split into two equal-size parts. For each part a word-map is computed, and then the two maps are concatenated and considered as a single pattern. It should be noted that in this case the size of the word-map is double than the case when the input sequence is not split. We refer to the case when the sequence is mapped into a single map (i.e. no splitting) as *one-matrix* map, and the case when the sequence is split in two halves as *two-matrix* map.

4. Experiments

In this section, we present studies where we evaluate how well the word-maps obtained with posterior features can discriminate between words compared to those obtained with cepstral features.

4.1. Database and features

We use the Phonebook speech corpus [9] for speaker- and task-independent small vocabulary isolated word recognition. Since, at this stage, we are not working yet at the final framework of our system, we use for our studies the cross-validation dataset. It consists of 8 subsets of utterances, each containing 75 words uttered on average by 11 speakers once. The speakers and the words present in the training data do not appear in cross validation set. There are 42 context-independent phones including silence. For more details about this dataset, the reader may refer to [10].

In our experiments, we split the cross validation data in two (roughly) equal-size sets. The first set contains 3618 word samples considered as templates; the second set contains 3672 samples used as test data. Again, it should be noted that the speakers present in the template set do not appear in the test data. We created several scenarios where the number of templates was varied from 1 up to 7.

In our experiments, we use two different kind of features:

- PLP features: 39-dimensional PLP cepstral features computed every 10ms of speech using window of 25ms.
- posterior features: 42-dimensional phone class-conditional posterior probabilities estimated using an *off-the-shelf* MLP trained on PLP features along with a temporal context of 90ms. This estimator was used in our previous work on posterior features [1].

4.2. SOM training

The SOMs used in our experiments have been trained using the SOM toolbox [11]. We trained SOMs with different feature map sizes, more specifically 10×10, 20×20, 30×30 and 40×40 nodes. For all the trained systems, the map had a rectangular structure and used a Gaussian neighbourhood function. The codebook vector of each node was initialized with uniformly distributed random values. For each size, we trained a SOM using PLP features and a SOM using posterior features. As discussed in Section 2.2, in case of cepstral features the standard SOM formulation using Euclidean distance as distance measure was used. In case of posterior features *dot-product* SOM was used.

4.3. Results

In this section we present the results obtained using different conditions, in particular using one-matrix maps (Figure 4a) and two-matrix maps (Figure 4b). In both cases, we compared the use of PLP cepstral features with the use of posterior features to obtain word-maps. Furthermore, for each kind of feature the systems have been tested on 7 scenarios (varying the number of templates per word) and 8 subsets. The performance reported in the figures for each system is measured as average accuracy over the 8 subsets.

The results show that posterior features significantly outperform PLP features. In previous works [5, 6, 7], cepstral features were used to obtain binary word-maps through SOM.

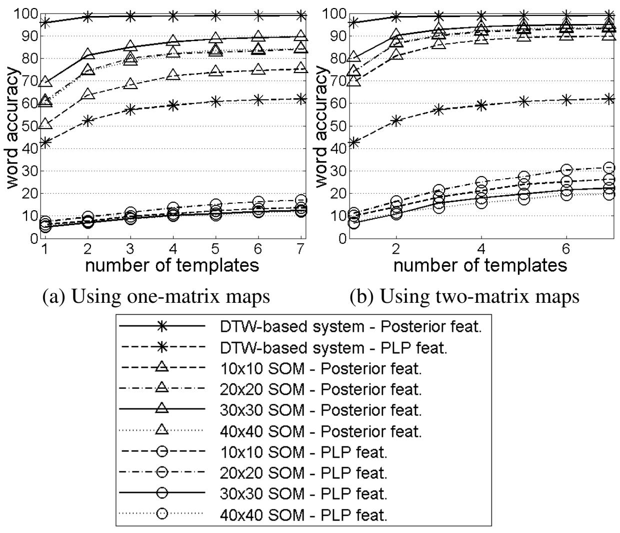


Figure 4: *Average recognition accuracy over the 8 subsets for each of the 7 scenarios (i.e. having different number of templates) using both PLP and posterior features. For posterior features the 30×30 map always perform better than the others, whereas 20×20 map is the best performing in case of cepstral features.*

These maps were then classified using classifiers such as MLP or SVM. In [5], the MLP approach was shown to yield performance of 99.5% accuracy on speaker-dependent task with 20-word vocabulary. In [6], it was shown to yield performance of 99.3% on speaker-independent task with 10-word vocabulary. In [7], the SVM approach on speaker-dependent 10-word vocabulary task yields up to 100% accuracy.

However, in this work we have a more challenging task with 75-word vocabulary, speaker-independent conditions and only few templates per word (which may not be enough to train a sophisticated classifier). Thus, the poor performance of PLP features in our experiments could be attributed to all these factors. This suggests that for PLP-based maps a sophisticated classifier may be needed while for posterior features a simple classifier could be sufficient.

Furthermore, it can be observed that the use of two-matrix maps yields better results than the corresponding scenario with one-matrix maps. This suggests that introducing temporal information improves the discrimination between words. It is interesting to notice that in case of posterior features the use of two-matrix maps reduces the difference in terms of performance between the various feature map sizes. Whereas, in case of PLP features the use of two-matrix maps increases the difference in the performance.

For a complete comparison, in the Figures 4a and 4b we also indicate the performance obtained for each scenario when a standard DTW-based recognition system is used for both PLP and posterior features. Comparing these two systems, it is evident that in case of PLP features the gap in the performance between DTW-based approach and SOM-based approach is wide. Even using two-matrix maps the distance between the performance of the two systems is still highly significant. In contrast, using posterior features and two-matrix maps the performance of SOM-based system approaches those of the DTW-based system.

5. Conclusions

In this work, we aimed at investigating the use of posterior features with SOM to produce fixed-size patterns representing words. We evaluated these patterns in the context of isolated word recognition. The results show that the maps obtained using posterior features are more sparse and easy to discriminate compared to those obtained using PLP features without the need of dynamic programming. This complements previous evidence [1, 2] that posterior features have interesting properties that could be exploited in new type of ASR systems. Along this direction, in our future work we intend to investigate the use of associative memory techniques which will replace the decision making step in Figure 1.

Also, in the context of posterior features it may be interesting to investigate the use of other distance measures, such as Kullback-Leibler divergence, in the formulation of SOM. In addition, it may be interesting to compare SOM with other clustering methods, such as K-means or Generative Topographic Mapping [12], to produce the same kind of patterns.

6. Acknowledgements

The authors would like to thank Prof. Hervé Bourlard for his insightful and critical comments and his fruitful discussions. This work was partly supported by the European Union under the Marie-Curie Training project SCALE, Speech Communication with Adaptive LEarning and the Swiss National Science Foundation under the National Centre of Competence in Research (NCCR) on Interactive Multimodal Information Management (IM2). More information about SCALE and IM2 is available from the project web sites www.scale.uni-saarland.de and www.im2.ch.

7. References

[1] S. Soldo, M. Magimai.-Doss, J. Pinto, and H. Bourlard, "Posterior Features for Template-based ASR," in *Proc. of ICASSP*, Prague, Czech Republic, 2011.

[2] A. Asaei, B. Picart, and H. Bourlard, "Analysis of Phone Posterior Feature Space Exploiting Class-Specific Sparsity and MLP-Based Similarity Measure," in *Proc. of ICASSP*, Dallas, Texas, USA, 2010.

[3] G. Palm, "On Associative Memory," *Biological Cybernetics*, vol. 36, pp. 19–31, 1980.

[4] T. Kohonen, *Self-Organizing Maps*, 3rd ed., ser. Springer series in information sciences, 30. Springer, 2001.

[5] Z. Huang and A. Kuh, "A combined Self-Organizing Feature Map and Multilayer Perceptron for Isolated Word Recognition," *IEEE Trans. on Signal Processing*, no. 40, pp. 2651–2657, 1992.

[6] P. Salmela, K. Laurila, S. Kuusisto, P. Haavisto, and J. Saarinen, "Isolated Spoken Number Recognition with Hybrid of Self-Organizing Map and Multilayer Perceptron," in *Proc. of ICNN*, Washington DC, USA, 1996, pp. 1912–1917.

[7] J. Manikandan and B. Venkataramani, "Study and evaluation of a multi-class SVM classifier using diminishing learning technique," *Neurocomputing*, vol. 73, pp. 1676–1685, 2010.

[8] T. Kohonen, "Self-organized Formation of Topologically Correct Feature Maps," *Biological Cybernetics*, vol. 43, pp. 59–69, 1982.

[9] J. Pitrelli, C. Fong, S. Wong, J. Spitz, and H. Leung, "Phonebook: A Phonetically-rich Isolated-word Telephone-speech Database," in *Proc. of ICASSP*, Detroit, Michigan, USA, 1995, pp. 101–104.

[10] S. Dupont, H. Bourlard, O. Deroo, V. Fontaine, and J.-M. Boite, "Hybrid HMM/ANN Systems for Training Independent Tasks: Experiments on 'PhoneBook' and Related Improvements," in *Proc. of ICASSP*, Munich, Germany, 1997, pp. 1767–1770.

[11] J. Vesanto, J. Himberg, E. Alhoniemi, and J. Parhankangas, "Self-Organizing Map in Matlab: the SOM Toolbox," in *Proc. of the Matlab DSP Conference*, 1999, pp. 35–40.

[12] C. M. Bishop, M. Svensén, and C. K. I. Williams, "GTM: The generative topographic mapping," *Neural Computation*, vol. 10, pp. 215–234, 1998.