A New HMM training and testing scheme

Albert Hung-Ren Ko
University of Toronto
105 St-George Street
Toronto, ON, Canada
drinkblue@gmail.com

Robert Sabourin
École de Technologie Supérieure
1100 Notre-Dame West Street
Montreal, QC, Canada
robert.sabourin@etsmtl.ca

Alceu de Souza Britto, Jr.
Catholic University of Parana
Imaculada Conceicao, 1155
Curitiba, Brazil
alceu@ppgia.pucpr.br

Abstract

One of disadvantages of Hidden Markov Models (HMMs) is its low resistance to unexpected noises among observation sequences. Unexpected noises in a sequence usually "break" a sequence of observations, and then makes this sequence unrecognizable for trained models. We propose a new HMM training and testing scheme, which compensates some of the negative effects of such noises. We carried out experiment on handwritten digit recognition problem and the result suggests our proposal can be as effective as multi-classifier systems.

1. Introduction

The purpose of handwritten pattern recognition systems is to achieve the best possible recognition rate, and the Hidden Markov Model (HMM) is one of the most popular classification methods for sequential pattern recognition problems [1, 4, 5]. The objective of the HMM is to model a series of observable signals, and it is this signal-modeling ability that makes HMMs a better choice than other classification methods for recognition problems. As a stochastic process, an HMM is constructed with a finite number of states and a set of transition functions between two states, or over the same state [1, 4, 5]. Each state emits some observation(s), according to a codebook setting out corresponding emission probabilities. Such observations may be either discrete symbols or continuous signals.

One important fact in the research concerning the HMM is that the ensemble of the HMM (EoHMM) emerges as a promising scheme for improving HMMs. This is because an ensemble of classifiers (EoC) is known to be capable of performing better than its best single classifier. However, even with EoHMMs, we see that the performance of HMMs is somehow more limited than in other types of classifiers. We believe that such a limited performance in some aspects might be attributed to the fundamental structure of HMMs, especially first-order HMMs. For first-order HMMs, all observations are supposed to stay at the same state or to move from current state to next state. This mathematical method allows us to match a given observation sequence with a model by examining the relationship between each current observation and its next observation.

Nevertheless, when there is unexpected noise in a sequence, the mathematical structure of an HMM will not enable us to discover the relationship between each current observation and its next observation. Thus, this sequence will become unrecognizable to trained models. We explain this phenomenon as a noise which "breaks" a sequence of observations. In fact, it is possible that more than one observation in a sequence might be noise. We face two types of problems: (a) unexpected noise occurring in a sequence of observations used for training; and (b) unexpected noise occurring in a sequence of observations used for testing. The former might bring about negative effects in a trained model, and the latter will make a sequence unrecognizable. It is difficult to handle these problems, since we have no information on whether or not an observation of a sequence is actually a noise.

To resolve these problems, we suppose that each observation has the same chance to be a noise. We thus propose a leave-one-out-training (LOOT-training) strategy: for a sequence of M observations, we leave an observation out of the original sequence and make a new sequence without this observation. We then apply the same process on all observations in this sequence and create M new sequences. We use all M sequences and the original sequence for training. For testing, we carry out the same process for each observed sequence, and create M new sequences from one original sequence for leave-one-out-testing (LOOT-testing). We sum up

978-1-4244-2175-6/08/$25.00 ©2008 IEEE

the likelihood of all new M sequences to make a final decision. Note that this is actually an example of a system with a single classifier and multiple classifications.

To verify our hypothesis, we constructed leave-one-out-training and leave-one-out-testing HMMs (LOOT-HMM), and carried out an experiment on the handwritten digit recognition problem. We used the HMM-based handwritten numeral recognizer in [2]. At this work, column and row HMM classifiers are used to enhance classification accuracy, and the sum of the outputs from the single best column HMM and the single best row HMM constitutes the final decision. With this system, we were able to improve verification by constructing a LOOT-HMM on both column HMM classifiers and row HMM classifiers. The paper is organized as follows: first, we introduce the basic concepts of LOOT-HMMs in section 2; in section 3, we report on experiments we carried out on the NIST SD19 handwritten numeral database. A discussion and a conclusion are presented in final sections.

2. Generation of LOOT-HMMs

The LOOT-HMM has three components: codebook optimization, LOOT-training, and LOOT-testing. We first explain the codebook optimization process using a clustering validity index, and then we explain the leave-one-out method for creating sequences of observations in HMM training and testing.

2.1 Codebook optimization using Xie-Beni (XB) Index

The first step in LOOT-HMM is codebook optimization. In general, an HMM codebook is generated from a vector quantization procedure, and each code word can be regarded as a centroid of a cluster in the feature space. The fitness of clustering depends on a number of different factors, such as clustering methods and the number of clusters. In this work, we focus on optimizing codebooks by finding the optimal number of clusters, i.e. by finding the optimal codebook size. In order to select the optimal number of clusters, We use the XB index [7] to scan the data and determine the number of clusters with the smallest XB index. This number is then chosen to be the codebook size, and codewords are generated using simple K-Means clustering [3]. After codebooks are created, we then apply the leave-one-out process for HMM training.

2.2 Leave-One-Out Training for HMMs

We use the Viterbi algorithm for standard HMM testing, but also implement a LOOT-testing process as a meta-testing. The leave-one-out process for HMM training is a simple one, which generates several sequences of observations from one sequence of observations. Suppose that, for any sequence S_i, we have the observations $o_1, o_2, \cdots, o_{M_i}$, where M_i is the number of observations for the sequence S_i. From this original sequence of observations, we then generate new sequences of observations by simply removing one observation at a time. The longer the original sequence, the more new sequences it will generate. LOOT-training gives us more training samples. Furthermore, it may reduce noise from, or induce noise into, a sequence. If it reduces the noise in a sequence, then HMMs have a more correct sample for training; if it induces a noise in a sequence, then HMMs can be trained to recognize noisy sequences in testing. LOOT-training is thus meant to make HMMs more robust. After LOOT-training has been completed, the HMMs are created and ready to classify any sequence tested. However, before doing this, we need to apply the LOOT-testing process on the sequences to be tested.

2.3 Leave-One-Out Testing for HMMs

LOOT-testing is similar to LOOT-training. The only difference is that, for testing, the leave-one-out process is applied on test sequences instead of on training sequences. Again, we generate new sequences of observations from the original sequence of observations by simply eliminating one observation at a time. After new sequences have been generated, all new m_i sequences and the original test sequence are submitted for testing. There are two schemes for generating final outputs based on these sequences. The first scheme (a) involves simply combining the likelihood of all sequences for each class, and select the class with the highest likelihood. This scheme uses a mechanism similar to the SUM rule in multi-classifier systems. The second scheme (b) involves directly selecting the class with the highest likelihood among all the classifications from all sequences, i.e. among all new m_i sequences and the original test sequence. Method (b) allows a sequence of observations to skip any single observation in order to achieve the highest likelihood. Then, by comparing the highest likelihood of each class, this method can assign the best class to the sequence by taking into consideration that one observation might be noise. We denote method (a) as LOOT-testing(SUM) and method (b) as LOOT-testing(MAX).

Also note that there are two premises in LOOT-testing. First, we suppose that we are unable to distinguish a noise from a normal signal in a sequence of observations. Thus, each observation is treated with an equal chance being a noise. Second, we suppose that a sequence is, in most cases, still recognizable without one of its observations. Based on these two premises, we can remove a single observation from a sequence of observations and submit it for classification. When we repeat this process, we submit a sequence of observations for classification multiple times by removing different observations. If the leave-one-out process takes away a true noise, then the new sequence has a better chance of being correctly classified. However, if the leave-one-out process takes away a normal observation, then all we can hope is that the new sequence will still be recognizable without the removed observation. The latter can be true when an omitted observation is not the only observation emitted by the current state. In other words, if the current state emits more than one observation, then the new sequence has a chance of being correctly recognized. In contrast, if the current state emits only one observation, i.e. the current state just moves from the previous state and will then move to the next state right away, then the new sequence might be unrecognizable. Nevertheless, if there are more recognizable new sequences than unrecognizable new sequences, the final classification result is likely to be correct. This might be the case when states are likely to undergo self-transition and emit more than one observation in a sequence. Therefore, if the number of observations in a sequence is significantly higher than the number of states, then the leave-one-out process might work well. To verify whether or not the leave-one-out process will work for HMMs, we perform some experiments on a benchmark database.

3 Experiments with HMMs

The experimental data were extracted from $NIST$ $SD19$ as a 10-class handwritten numeral recognition problem. The training set with $150,000$ samples ($hsf_\{0-3\}$) was used to create 2 HMM classifiers, one being the column HMM classifier and another being the row HMM classifier. Due to the extremely large dataset, we use only the first $10,000$ images from the training dataset to evaluate the quality of the clustering, and they are equal to $342,910$ columns and $461,146$ rows. However, codebooks were generated with the whole training set. The training validation set of $15,000$ samples was used to stop HMM training once the optimum had been achieved. Using the best solution from this archive, the test set containing $60,089$ samples ($hsf_\{7\}$) was used to evaluate the accuracy of EoCs.

Table 1. The performance of column LOOT-HMMs and row LOOT-HMMs.

	Column	Row	Combination
Original	97.60%	96.76%	98.00%
Optimization	98.66%	97.78%	98.79%
LOOT-Traing	98.69%	98.22%	98.86%
LOOT-Testing(SUM)	98.68%	98.15%	98.86%
LOOT-Testing(MAX)	98.74%	98.15%	98.85%
LOOT-Traing+Testing(SUM)	98.72%	98.14%	98.84%
LOOT-Traing+Testing(MAX)	98.76%	98.23%	98.88%

Each column HMM used 47 features obtained from each column, and each row HMM used 47 features obtained from each row. For benchmark column HMM, we have a recognition rate of 97.60%, and for benchmark row HMM, the classification accuracy was about 96.76%, and the combination of benchmark column HMM and benchmark row HMM achieved a rate of 98.00%. Note that the combination of column HMM and row HMM is simply the combination of likelihoods of column HMM and row HMM for each class, and the class with the highest combined likelihood is chosen as the final output.

3.1 Optimization for HMMs

To decide on suitable codebook sizes for HMMs, we carried out clustering analyses on HMM features. We scanned the data with the number of clusters from $1 \sim 10,000$. For column HMMs, we determined that the optimal codebook size is 9525; for row HMMs, the best codebook size is 9783. After the codebook optimization process, we carried out the optimization of the number of states [6].

3.2 Column LOOT-HMMs and Row LOOT-HMMs

The optimization of codebooks and the numbers of states can improve the accuracy of column HMMs from 97.60% to 98.66%, and that of row HMMs from 96.76% to 97.78%. Surprisingly, the use of leave-one-out training or leave-one-out testing does not greatly improve the accuracy of column HMMs. By contrary, their use yields more improvements on row HMMs (Table 1).

When we combine LOOT-Training-Testing column HMMs and LOOT-Training-Testing row HMMs, we achieve a 98.88% recognition rate on handwritten numerals. The performance is good compared with the

recognition rate of 98.00% of the original combination of column and row HMMs. Note, however, that if we skip the LOOT-training process and only carry out LOOT-testing after optimization, we can still achieve a 98.86% recognition rate, while LOOT-training without LOOT-testing(MAX) can also achieve a recognition rate of 98.86%. Again, this confirms an overlapping effect from LOOT-testing and LOOT-training. Also note that the best result, 98.88%, is only slightly better than the best results that can be achieved with EoHMMs for the dataset in question. This might hint that a meta-learning approach - either with multiple classifiers or with multiple classifications - has its own limits. Further improvement should focus on the learning structure of the HMM, and not on meta-learning methodologies.

4 Discussion

The proposed LOOT-HMM is an easy and relatively effective scheme. The column HMMs achieve a 98.76% recognition rate, while the row HMMs have a 98.23% accuracy. More importantly, the combination of column and row HMMs improves performance, which reaches 98.88%.

LOOT-testing offers improvement similar LOOT-training. The LOOT-testing process uses a single classifier, but can generate multiple classifications. By using these multiple classifications, LOOT-testing might offer similar advantages to those of an EoC. For example, an EoHMM achieves 98.86% [3] on the same dataset using a pool of 40 classifiers, i.e. 400 HMMs. Compared with the EoHMM scheme, our LOOT scheme uses only two classifiers, i.e. 20 HMMs, and thus costs less and has a similar performance. It might therefore be of great interest to further pursue this direction. Note that the use of LOOT-training alone seems to have effects similar those of LOOT-testing. Without LOOT-testing, LOOT-training can still a achieve 98.86% recognition rate, and vice versa. Note that if only LOOT-testing is implemented, then LOOT-HMM will have a time-saving advantage over EoHMM, because only one HMM classifier training is required.

5 Conclusion

In general, HMMs are constructed using a structure in which each state emits some observable symbols, and one state moves on to another state with a certain probability. From this, we obtain the probability of emission of each symbol from each state and the probabilities of transition among states, and then establish models. Finally, observation sequences are matched to these trained models. However, the fundamental structure of an HMM implies that the models have little resistance to unexpected noise among observation sequences. To take into account the effect of unexpected noise occurring in a sequence of observations, we propose a leave-one-out methodology for HMM training and HMM testing. We find that LOOT-training could make HMMs slightly more robust, and LOOT-testing could enhance the accuracy of HMMs. This is especially significant when there are longer sequences of observations, such as in row HMMs.

Our study also raises more questions, such as how to guarantee the optimality of the leave-out process, since the number of possible scenarios for this process is very large. The leave-one-out strategy presented in this paper is straightforward, however further improvements can be made. We believe that this methodology can be greatly enhanced with theoretical studies on the distinguished nature between the observations and the potential noise, the influence of geometrical and topological constraints on the HMMs, better statistical studies to quantify the uncertainty of the nature of observations, and empirical studies on more real-world problems.

References

[1] Y. Bengio Markovian Models for Sequential Data *Neural Computing Surveys*, vol. 2, pp. 129-162, 1999

[2] A. Britto, R. Sabourin, F. Bortolozzi and C.Y. Suen Recognition of Handwritten Numeral Strings Using a Two-Stage Hmm-Based Method *International Journal on Document Analysis and Recognition (IJDAR 5)*, no. 2-3, pp. 102-117, 2003

[3] A.H.R. Ko, R. Sabourin, and A. Britto Jr. A New HMM-Based Ensemble Generation Method for Character Recognition *Workshop on Multiple Classifier Systems*, pp. 52-61, 2007

[4] L. R. Rabiner A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition *Proceedings of the IEEE*, 77(2):257 – 286, 1989

[5] P. Smyth, D. Heckerman and M. I. Jordan Probabilistic independence networks for hidden Markov probability models *Neural Computation*, vol. 9, pp. 227–269, 1997

[6] X. Wang Durationally constrained training of HMM without explicit state durational *Proceedings of the Institute of Phonetic Sciences 18*, pp. 111-130, 1994

[7] X. L. Xie and G. Beni A validity measure for fuzzy clustering *IEEE Transactions of Pattern Analysis and Machine Intelligence*, pp. 841-847, 1991