Multiple Sequence Alignment Based Bootstrapping for Improved Incremental Word Learning

Irene Clemente, Martin Heckmann, Gerhard Sagerer, Frank Joublin

2010

Preprint:

This is an accepted article published in International Conference on Acoustics, Speech, and Signal Processing (ICASSP). The final authenticated version is available online at: https://doi.org/[DOI not available]

MULTIPLE SEQUENCE ALIGNMENT BASED BOOTSTRAPPING FOR IMPROVED INCREMENTAL WORD LEARNING

Irene Ayllón Clemente[1,2], *Martin Heckmann*[2], *Gerhard Sagerer*[1], *Frank Joublin*[2]

[1]Research Institute for Cognition and Robotics
Bielefeld D-33615, Germany
{*iayllon,sagerer*}*@cor-lab.uni-bielefeld.de*

[2] Honda Research Institute Europe GmbH
Offenbach D-63073, Germany
{*firstname.lastname*}*@honda-ri.de*

ABSTRACT

We investigate incremental word learning with few training examples in a Hidden Markov Model (HMM) framework suitable for an interactive learning scenario with little prior knowledge. When using only a few training examples the initialization of the models is a crucial step. In the bootstrapping approach proposed, an unsupervised initialization of the parameters is performed, followed by the retraining and construction of a new HMM using multiple sequence alignment (MSA). Finally we analyze discriminative training techniques to increase the separability of the classes using minimum classification error (MCE). Recognition results are reported on isolated digits taken from the TIDIGITS database.

Index Terms— Speech recognition, Hidden Markov models, training, sequence estimation

1. INTRODUCTION

Current speech recognition systems are trained offline with large databases. However, children learn language in the interaction with their caregivers and the environment. Our goal is to model this process via an interactive learning scenario where a human tutor teaches a robot.

Most automatic speech recognition systems (ASR) use Hidden Markov Models. With conventional training techniques, a large amount of labelled training data is needed to estimate an optimal set of parameters. Unfortunately, it is very difficult to obtain this in interactive learning; hence researchers aim either to train the system in an unsupervised manner or to train it with a smaller number of training data. In both cases, the efficiency of the procedure strongly depends on the initialization of the parameters. In these conditions, a bootstrapping phase is required to get a good set of model parameters.

Different approaches for initializing Hidden Markov Model parameters exist. In the standard approach, proposed in [1], each training segment is considered as the output of the HMM whose parameters are to be estimated. Hence, if the state that generated each frame or observation vector in the training segment was known, then the means and variances of the model could be computed by averaging all the vectors associated with each state. This is achieved by segmenting the training data frames into states. This segmentation is performed via K-means clustering and Viterbi decoding. The initialization of the Hidden Markov Model parameters in HTK [2], HINIT, is based on this technique. The procedure is realized in a supervised way requiring labelled training data. Nevertheless, this kind of approach only ensures optimal convergence if many labelled training samples are available, which is not the case in incremental learning.

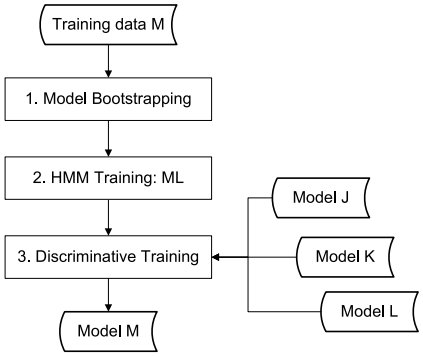

Fig. 1. Incremental discriminative training system. ML stands for Maximum Likelihood estimation.

Similar supervised initialization techniques to the ones described above can be found in the literature (see [3] as example). In contrast to this, several authors have been focused on unsupervised and online learning in interactive environments. As illustration, an unsupervised phone model acquisition procedure is proposed in [4] and [5]. This technique, based also on K-means clustering and Viterbi decoding, enables the initialization of syllable models in an online word learning system [6] using the previously estimated phone models. Both methods are also able to work with few samples.

In this paper we propose an incremental word learning framework with few training data as depicted in Figure 1. This technique consists of three main steps. First, a novel model bootstrapping method initializes the parameters of the Hidden Markov Models. It consists of the combination of unsupervised and supervised training, where a transformation of an ergodic HMM into a left-to-right HMM takes place. This transformation is performed by means of a novel multiple sequence alignment procedure. This first step is the main contribution of this paper and allows to estimate a good initial set of HMM parameters, which are trained by the Baum Welch algorithm [1] in the next phase. Minimum classification error (MCE) training refines the estimates of the parameters computed in the step before. It reduces the classification error in the training data, allowing a better separation of the classes.

The rest of the paper is organized as follow. In Section 2 we give an overview of our model bootstrapping, describing in more detail the different phases in consecutive subsections. The discriminative training technique used is presented in Section 3. In Section 4 we report results for our approach on an isolated digit recognition task and compare them to standard approaches. Finally, in Section 5 we discuss the results and give an outlook on future work.

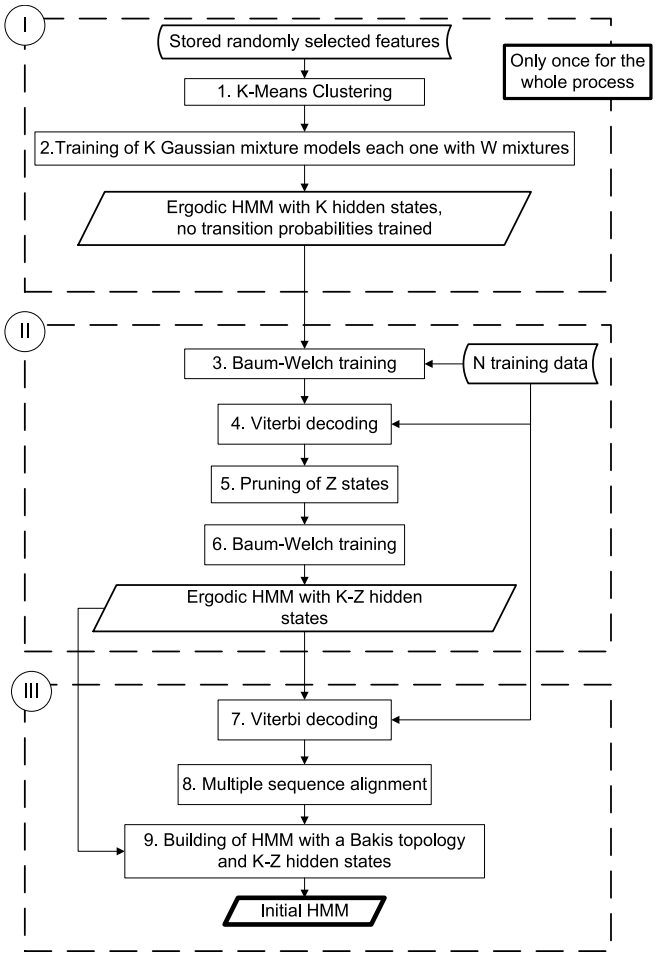

Fig. 2. Overview of the proposed model bootstrapping system to initialize the HMMs. It consists of three main parts and nine processing steps. Part I is explained in section 2.1, part II in section 2.2 and part III in section 2.3.

2. MODEL BOOTSTRAPPING

Hidden Markov Models are usually trained by means of the Baum-Welch algorithm [1], which is based on the maximum likelihood (ML) criterion. Unfortunately, the Baum-Welch algorithm easily gets stuck in local minima. Thus, it is essential to have a model bootstrapping which provides an adequate initialization for the HMM parameters to obtain good convergence.

The proposed model bootstrapping system is shown in Figure 2. This algorithm comprises three main stages: the unsupervised training of a generic HMM (very similar to [4]), in which a common HMM initialization model is constructed without using any labelled training data. Next, training of the previously obtained HMM using the Baum-Welch algorithm [1] and labelled training data is performed. This yields ergodic word-level HMMs. The topology of the HMM obtained is transformed into a Bakis, i.e. left-to-right, configuration by means of a novel multiple sequence alignment. These steps form the basis for the construction of a new word-level HMM.

2.1. Unsupervised training of a generic HMM

The top of the scheme displayed in Figure 2 as part I is based on the approach proposed in [4] and [5]. To initialize the procedure, a few minutes of input speech are recorded. Acoustic features are extracted and clustered by the K-Means algorithm. In [4] the resulting K clusters are used to train single-state continuous HMMs. However, in our approach an ergodic Hidden Markov Model with K hidden states is trained instead of K single-state HMMs. In the process described, training is performed in a completely unsupervised manner and it is only executed once. The resulting HMM is stored to be used as pre-initialization for each new model to create.

2.2. Training of the word model

Next, the ergodic HMM is retrained by means of the Baum-Welch algorithm using a labelled training data set. In our bootstrapping technique, we use a predefined number of states. Hence, a state pruning similar to the one described in [6] is performed. A Viterbi decoding of the training segments is realized and then the least occupied states are pruned. The observation estimates of the states are not changed, and the transition matrix is constructed by eliminating the rows and columns belonging to the pruned states. A further Baum-Welch training refines the estimates.

2.3. Multiple sequence alignment

For isolated word recognition, the configuration of states has to be changed into a Bakis, i.e. left-to-right, topology. The idea behind the Bakis-topology is that transitions between states are only forwards, i.e. from left to the right. Hence it is advisable to perform a Viterbi decoding of the training segments to obtain the most likely state-sequence generating the data. This results in N optimal path sequences one for each of the N training samples per word. Each sequence contains information about a possible underlying configuration of states in a left-to-right topology for its corresponding training segment. Thus, all sequences have to be merged into one sequence which codes the information contained in the Viterbi decoding sequences. This is performed via a multiple sequence alignment algorithm.

The term sequence alignment is used in Bioinformatics to define a way of arranging different biological sequences to identify similar regions that may be an indication for some kind of relationship between the sequences. Over the last decades, several efficient algorithms have been developed in order to align protein and gene sequences [7], [8]. The goal of these algorithms is to find an alignment, which is optimal under a scoring function. In most cases, the scoring function is provided with a similarity matrix. This matrix assigns costs for the replacement of one element by a different one. The scoring function is built in a way such that the best alignment is only to be expected if both sequences are the same.

Our goal is to merge all the sequences in a succession of states representing the best alignment between the sequences. The alignment of multiple sequences has also been studied in Bioinformatics (see [9]). The multiple sequence alignment method we propose is displayed in Figure 3, which consists of three main building blocks:

The first block is the calculation of a cost matrix D assigning different costs to the permutation of state transitions in the observed state sequences. This cost matrix is analogous to the similarity matrix used in Bioinformatics. However, they differ in the computation of the probability coded in each element of the matrix. Each element of the cost matrix represents the probability of an element to be followed by another, different, element. The element $D(i, j)$ of the

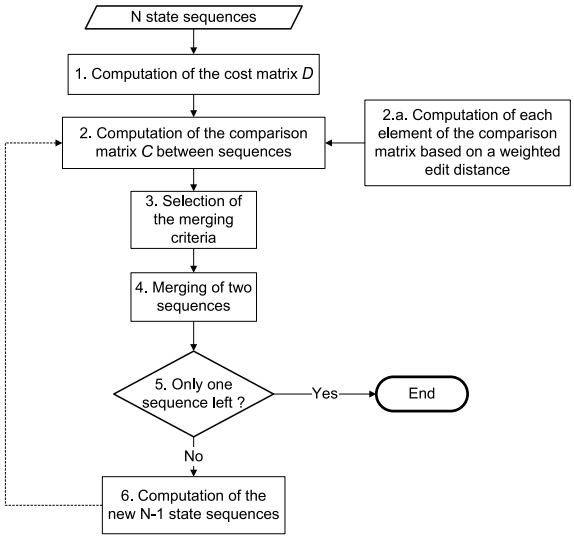

Fig. 3. Multiple sequence merging procedure.

cost matrix D is calculated based on the frequency of the sequence $j \rightarrow i$ in the sequences following the Viterbi decoding (see Equation 1). This differs from approaches in Bioinformatics where similarity matrices represent the probability of a character to be aligned by another one.

$$D(i,j) \sim \frac{\sum_{seq} \delta_{j \rightarrow i}}{\sum_i \sum_{seq} \delta_{j \rightarrow i}} \tag{1}$$

The second main building block is the calculation of the distances between all sequences, captured by the comparison matrix C. The distances between two sequences are computed by means of a special weighted edit distance using dynamic programming. This is computed as a result of the construction of a distance matrix S (see Equation 2) which can be considered as a modified fusion of the H-Matrix of the algorithm proposed in [8] and the matrix proposed in [7], often referred to as F-Matrix. In Equation 2, $c(i,j)$ is a value depending on the alignment of the elements i and j of the sequences to compare v and t and the cost matrix D (see Equation 3).

$$\begin{gathered} S(1,j) = 0 \;\; ; \;\; S(i,1) = 0 \\ S(i,j) = max[S(i,j-1), S(i-1,j-1)] + c(i,j) \end{gathered} \tag{2}$$

$$\begin{gathered} a = v(i-1) \\ b = t(j-1) \\ \\ c(i,j) = \begin{cases} 1 & if \quad a = b \\ D(b,a) & if \quad a \neq b \end{cases} \end{gathered} \tag{3}$$

Finally, in the third block different sequences are merged based on their similarity measure until only one sequence is left. In our approach we start merging the least similar sequences. The merging procedure is a forward decoding of the matrix S, used to compute the weighted edit distance referred to before. This merging procedure is similar to the backtracing proposed in [8]. However, in our approach the walk is forwards. After merging all sequences, we obtain one optimal sequence.

Once the succession of states is computed, the new HMM with Bakis topology is constructed. The Gaussian mixture parameters are conserved from the previous steps, however the transition matrix of this new HMM with Bakis topology is initialized with the values calculated by the cost matrix D.

3. DISCRIMINATIVE TRAINING

As referred to in section 1, HMMs are usually trained via the Baum-Welch algorithm [1] in a ML fashion. This estimation method tries to fit the statistical models (HMMs) best to the training data. However, the resulting distributions generally differ from the true distribution of the speech segments and with few training examples it is not possible to get a reliable estimation. Hence the theoretical minimum classification error, also called Bayes error, can not be achieved.

Instead of ML estimation, discriminative training (DT) has also been widely studied for HMMs in ASR [10],[11]. The DT methods aim to directly minimize or reduce classification errors in training data as model estimation criterion. Minimum classification error (MCE) is currently one of the prominent DT approaches. In our system, the minimum classification error estimation using the extended Baum-Welch (EBW) algorithm proposed in [12] is performed.

As shown in Figure 1 after training the HMMs by ML, an incremental minimum classification error technique referenced in Figure 1 by item 3 is realized. To model incremental word learning, we start the MCE optimization with only one class. When a new word model is trained the previously constructed models are used to increase the separation between them by means of computing the MCE technique described in [12]. Here only the new word model is updated. In the previous ML based steps there is no difference between incremental and batch learning.

4. EXPERIMENTS

4.1. Experimental procedure

We have evaluated our incremental word learning system using a subset of the TIDIGITS corpus [13]. The subset of the corpus selected contains utterances from 112 men collected from 21 regions of the United States. There are a total of eleven words (digits) in the corpus vocabulary (digits of "1" to "9", plus "oh" and "zero"). Each utterance is an isolated-digit string. The test set contains 112 samples for each digit and the training set up to 10 repetitions for each digit, each segment being uttered by a different speaker. In our experiments, all data is sampled at a rate of 16 KHz. The 45-dimensional acoustic feature vectors are composed of 15 RASTA-PLP coefficients [14] and their first and second order time derivatives. The models used in our experiments are continuous density HMMs (CDHMMs) with a fixed number of 3 Gaussian mixture models for each state; the number of states is fixed to 16. The models are trained first using the model bootstrapping explained in section 2. The baseline system used is a Hidden Markov Models framework implemented as in [1] using a statistical Matlab Toolbox called NETLAB [15].

First we compare the proposed bootstrapping with a conventional initialization as described in [1], the baseline system. Next, we want to compare our multiple sequence alignment (MSA) based bootstrapping with a simpler bootstrapping approach. In [6] an approach we want to refer to as best Viterbi alignment (BVA) is proposed. It is based on a very similar bootstrapping as our approach followed by a Viterbi decoding for all training samples (compare Figure 2 step 7). In BVA instead of aligning and merging all sequences the state sequence with the highest probability for all training data is chosen. As we set the number of states for each HMM to 16 also in this model, a pruning of the least occupied states is realized.

No. data	Baseline Mean	Min	Max	**MSA** Mean	Min	Max	BVA Mean	Min	Max
10	1.1	0.3	2.3	0.4	0.1	1.0	0.4	0.1	0.8
9	1.6	1.0	2.8	0.7	0.6	1.2	0.5	0.3	0.8
8	2.3	1.2	4.4	1.0	0.6	1.4	0.9	0.6	1.1
7	3.0	1.2	6.4	1.3	0.9	1.9	1.3	0.7	1.7
6	4.3	2.3	7.6	1.8	1.1	3.4	1.9	0.9	3.3
5	6.9	4.1	12.3	2.9	1.5	4.8	3.3	2.0	5.1
4	9.3	4.9	14.7	4.5	1.8	6.7	5.2	2.9	7.4
3	12.9	7.9	18.3	6.4	3.3	9.8	7.6	5.4	10.9
2	16.7	12.7	21.3	8.6	6.8	11.5	10.7	8.2	16.5
1	23.9	16.1	32.9	14.3	10.1	25.2	14.3	10.1	25.2

Table 1. Word Error Rates (WER %) of the different model bootstrapping methods compared to the baseline system. For each method, the WER value of the first column represents the mean of a 10-fold speaker-independent cross-validation on the training data set. The second and third column are the minimum and maximum of the cross validation respectively. MSA stands for the here proposed multiple sequence alignment bootstrapping method and BVA for the best Viterbi alignment approach proposed in [6].

4.2. Experimental results

In comparison to the baseline system the BVA approach in [6] and our proposed MSA bootstrapping method described in section 2 distinctly reduce the word error rates (WER) (compare Table 1). Furthermore, our model bootstrapping technique MSA achieves similar or better results compared to BVA. Especially, for a very small number of training samples (2-6) MSA is superior to BVA. When only one training segment is used, the recognition results obtained in our approach and in [6] are the same. In this case the multiple sequence alignment algorithm described in section 2.3 is not executed, because only one training segment is sampled by Viterbi decoding. At this point our technique and [6] have a very similar behaviour.

The execution of the incremental minimum classification error technique did not further improve the recognition results of the system.

5. DISCUSSION & SUMMARY

We have proposed an incremental word learning system improved by a new model bootstrapping approach. The key concepts of our approach include the combination of unsupervised and supervised training and the incorporation of a novel multiple sequence alignment technique initially applied in Bioinformatics. We have evaluated our method in an isolated digit recognition task using a subset of the TIDIGITS database. This is a very simple task. Hence, the model bootstrapping technique that we have developed to initialize the HMM parameters needs to be validated with more complex tasks in order to verify our approach. However, we have shown very promising results for training with very few samples. We could demonstrate (see Table 1) that our system is able to outperform a baseline system and a previously presented incremental word learning approach [6] when a very small number (2-6) of training segments is used. In interactive learning, it is not desirable that a human tutor needs more than 3-4 repetitions to teach a word to a robot. Thus, the proposed model bootstrapping system provides encouraging results for our targeted application. Moreover, it is a substantial improvement that using only two training samples the recognition error is reduced to 8.6 % WER in a speaker-independent task. The application of incremental MCE training was not beneficial because of the very small number of training data used. In this case, after ML training all training samples were already classified correctly. Hence, our efforts are currently focused on integrating large-margin minimum classification error techniques, which would improve recognition results also when no training errors occur.

6. ACKNOWLEDGEMENTS

We want to thank Christian Goerick, Tobias Rodemann and Britta Wrede for fruitful discussions.

7. REFERENCES

[1] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–286, 1989.

[2] S.J. Young, G. Evermann, M.J.F. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P.C. Woodland, *The HTK Book, version 3.4*, Cambridge University Engineering Department, Cambridge, UK, 2006.

[3] K. Nathan, A. Senior, and J. Subrahmonia, "Initialization of hidden markov models for unconstrained on-line handwriting recognition," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Process.*, 1996, pp. 3502–3505.

[4] H. Brandl, B. Wrede, F. Joublin, and C. Goerick, "A self-referential childlike model to acquire phones, syllables and words from acoustic speech," in *7th IEEE Int. Conf. Development and Learning.*, 2008, pp. 31–36.

[5] N. Iwahashi, "Robots that learn language: Developmental approach to human-machine conversations," in *Symbol Grounding and Beyond: Proc. of the 3rd Int. Workshop on the Emergence and Evolution of Linguistic Communication*, P. Vogt and et al., Eds. 2006, pp. 143–167, Springer.

[6] H. Brandl, *A computational model to unsupervised childlike speech acquisition*, Ph.D. thesis, Bielefeld University, 2010.

[7] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, March 1970.

[8] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences.," *Journal of Molecular Biology*, vol. 147, no. 1, pp. 195–197, March 1981.

[9] J.D. Thompson, D.G. Higgins, and T.J. Gibseon, "CLUSTALW: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice," *Nucl. Acids Res.*, vol. 22, pp. 4673–4680, 1994.

[10] B-H. Juang, W. Hou, and C-H. Lee, "Minimum classification error rate methods for speech recognition," in *IEEE Trans. on Speech and Audio Process.*, 1997, pp. 257–265.

[11] E McDermott, *Discriminative training for speech recognition*, Ph.D. thesis, Waseda University, 1997.

[12] X. He, L. Deng, and W. Chou, "Discriminative learning in sequential pattern recognition - a unifying review for optimization-oriented speech recognition," *IEEE Signal Processing Magazine*, vol. -, pp. 14–36, 2008.

[13] R. Leonard, "A database for speaker-independent digit recognition," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Process.*, 1984, pp. 328–331.

[14] H. Hermansky and N. Morgan, "Rasta processing of speech," *IEEE Trans. Speech and Acoustics*, vol. 2, pp. 587–589, October 1994.

[15] I. T. Nabney, *NETLAB: algorithms for pattern recognition*, Springer Advances in Pattern Recognition Series, 2002.