Available online at www.sciencedirect.com

SciVerse ScienceDirect

Computer Speech and Language 27 (2013) 969–988

COMPUTER SPEECH AND LANGUAGE

www.elsevier.com/locate/csl

Joint training of non-negative Tucker decomposition and discrete density hidden Markov models

Meng Sun *, Hugo Van hamme

Department of Electrical Engineering-ESAT, Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, Bus 2441, B-3001 Leuven, Belgium

Received 7 February 2012; received in revised form 26 September 2012; accepted 28 September 2012
Available online 18 October 2012

Abstract

Non-negative Tucker decomposition (NTD) is applied to unsupervised training of discrete density HMMs for the discovery of sequential patterns in data, for segmenting sequential data into patterns and for recognition of the discovered patterns in unseen data. Structure constraints are imposed on the NTD such that it shares its parameters with the HMM. Two training schemes are proposed: one uses NTD as a regularizer for the Baum–Welch (BW) training of the HMM, the other alternates between initializing the NTD with the BW output and vice versa. On the task of unsupervised spoken pattern discovery from the TIDIGITS database, both training schemes are observed to improve over BW training in terms of pattern purity, accuracy of the segmentation boundaries and accuracy for speech recognition. Furthermore, we experimentally observe that the alternative training of NTD and BW outperforms the NTD regularized BW, BW training and BW training with simulated annealing.
© 2012 Elsevier Ltd. All rights reserved.

Keywords: Non-negative Tucker decomposition; Hidden Markov models; Unsupervised training; Regularization; Speech recognition; Sequential pattern discovery; Vocabulary acquisition

1. Introduction

Hidden Markov models (HMM) are good at modeling sequential data and have thus been applied to a lot of tasks of sequential data processing, such as automatic speech recognition (ASR) (Rabiner, 1989), topic detection and segmentation (Blei and Moreno, 2001), handwriting recognition (Hu et al., 1996) and gene analysis (Pedersen and Hein, 2003). To estimate the unknown parameters of an HMM, usually the likelihood of the data is maximized with the expectation maximization (EM) method (Baum et al., 1970; Juang and Rabiner, 1990; Wu, 1983). Often, sequential data comes with sequential labels, resulting in a *supervised* training problem. For instance, in state-of-the-art ASR systems, the training data is labeled in terms of the words or phones and an HMM representation is learned for each label.

In some tasks the labels are not provided and the training is therefore *unsupervised*. For example, unsupervised vocabulary discovery aims to learn word-like sequential patterns from input speech data in a completely data-driven way. The motivation for studying unsupervised vocabulary discovery is fourfold. Firstly, it can serve to model early language acquisition of infants where they acquire spoken units from exposure to the spoken language. These units are

* Corresponding author. Tel.: +32 016 32 18 30; fax: +32 016 32 17 23.
E-mail addresses: mengsun@esat.kuleuven.be, sunmengccjs@gmail.com (M. Sun), hugo.vanhamme@esat.kuleuven.be (H. Van hamme).

0885-2308/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.csl.2012.09.006

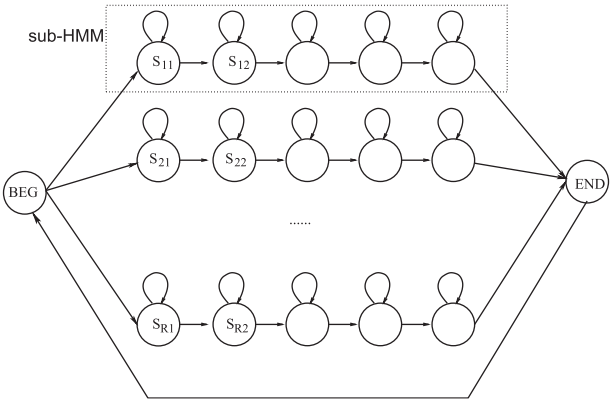

Fig. 1. The HMM topology used for sequential pattern discovery. Each parallel branch is called a sub-HMM.

subsequently related to multi-modal observations (e.g. images, actions) which is a form of weak supervision (Boves et al., 2007; Roy, 2003; Heckmann et al., 2009; Clemente et al., 2010). Secondly, a similar learning problem is created when one wants to communicate with robots in natural language to assign them a task. Such instructions are bound to contain environment-specific vocabularies that the robot needs to learn from interaction with humans. Thirdly, unsupervised learning of HMMs does not require transcriptions of speech, so the huge human effort to generate hand labels is avoided (Park and Glass, 2008; Zhang and Glass, 2010). Finally, unsupervised vocabulary acquisition can be applied in modeling out-of-vocabulary (OOV) words (ten Bosch et al., 2008; Jansen et al., 2010) and in the analysis of speech in underresourced languages or dialects (Zhuang et al., 2009).

In sequential pattern discovery, the only information that is available to us is that the data contains recurring *sequential patterns* showing some variation and possibly embedded in data that does not show much structure. In this article, it is assumed that each sequential pattern is adequately modeled by a sub-HMM, i.e. sequential pattern discovery is cast as the unsupervised estimation of the parameters of an HMM with the topology of Fig. 1. The objective function of unsupervised learning is usually the data likelihood which is maximized by using numerical optimization methods such as gradient descent algorithm (Levinson et al., 1982) or EM methods via optimizing an auxiliary function such as in Baum–Welch training (Baum et al., 1970; Khreich et al., 2012). Since EM is easier to apply than the gradient-based numerical optimization and is suitable for solving problems with a large number of parameters, it has been vastly adopted in ASR. Though the EM algorithm guarantees non-decrease of the data likelihood at each iteration, it can converge to local extrema due to the non-convexity of the optimization problem. Even in *supervised* learning careful initialization procedures with gradual increase of the model and data complexity are required to yield HMM parameter estimates that result in accurate recognition scores (Huang et al., 2001). In *unsupervised* learning, such procedures are more difficult to implement by the very nature of the problem: the data is unlabeled and its content is unknown. When applied blindly, the EM algorithm will produce poor local optima, as was observed in Smith (2006) and Johnson (2007) in discovering linguistic structure or as will be shown in Section 5.2.3 on word discovery where patterns corresponding to multiple words are found as illustrated in Fig. 5. Thus the unsupervised learning problem requires other solutions than supervised learning.

Our ultimate goal is pattern discovery in unlabeled data. In doing so, the avoidance of local optima in EM training emerges as an important issue which can be addressed with simulated annealing, i.e. perturbing the parameter estimates randomly with a magnitude that decreases with the iteration number (Paul, 1985). Simulated annealing is claimed to be able to yield the global optimum with probability one given *sufficiently slow* annealing and *enough* iterations. In practice, these conditions are hard to satisfy. In the example given in Paul (1985), a small HMM system with 8 states and 9 observation symbols required 60,000–400,000 iterations to attain high data likelihoods. On the other hand, we should notice that the target of sequential pattern discovery is not only to find a group of solutions with high data likelihood, but also to discover meaningful sequential patterns in the data. Hence, a model to jointly learn ground truth labels and HMM parameters is proposed in Siu et al. (2011). The iterative training procedures guarantee the non-decrease of the joint objective function. However, experiments reported in this article will show that the method is affected by the choice of initial ground truth labels which are treated as "correct" supervision in the following EM

training. In Jansen and Church (2011), HMMs are built unsupervisedly by first discovering repeated acoustic patterns, called *word clusters*, by segmental dynamic time warping (SDTW) (Park and Glass, 2008; Zhang and Glass, 2010; Jansen et al., 2010), then training HMMs for each word cluster and finally clustering HMM states across words. Other relevant work relates to the unsupervised training of HMMs for sub-word units (Varadarajan et al., 2008; Driesen et al., 2012). Their approach is to gradually increase model complexity using well-engineered methods for clustering and splitting states modeled by multivariate Gaussians, with the goal of discovering *meaningful* sub-word units. With the Viterbi alignment between the yielded state sequences and the ground truth labels, the methods are able to model word-like spoken patterns. By contrast, we start from an HMM topology for which the parameters are estimated from the data by exploiting a tensor factorization to force the HMM to meaningful solutions. In other words, an important difference with the current work is that a tensor factorization method takes over the role of the SDTW in Park and Glass (2008), Zhang and Glass (2010) and Jansen et al. (2010) and the role of state splitting and clustering in Varadarajan et al. (2008) and Driesen et al. (2012) and is incorporated in the iterative process of (discrete density) HMM parameter estimation.

In this article, we propose an approach to training that brings the following ideas together.

1.1. Co-occurrence-based unsupervised HMM learning

Recently, alternatives to unsupervised EM training of discrete density HMMs based on the *joint distribution of consecutive observation variables* have been proposed (Hsu et al., 2011; Song et al., 2010; Lakshminarayanan and Raich, 2010; Yang and Oja, 2011; Cybenko and Crespi, 2011; Finesso et al., 2010; Vanluyten et al., 2008). Instead of maximizing the likelihood of the observation sequences directly, they rely on the fact that the joint probability of two observation variables is structured by the HMM (see Eq. (8)).The joint distribution of consecutive observation variables is estimated by the statistics of *co-occurrences of observable symbols*, i.e. the accumulated counts of paired observable symbols in the training data. The co-occurrences of observable symbols are arranged into a so-called *co-occurrence matrix* with a number of rows and columns equal to the size of the observable symbol set. Conversely, the HMM structure can be estimated from the joint probabilities by a matrix factorization. We classify the models into two categories:

(i) Spectral embedding via singular value decomposition (SVD).

In Hsu et al. (2011) and Song et al. (2010), it is shown that symbol co-occurrence probabilities of discrete-density HMMs show structure that can be recovered using an SVD. The yielded eigenvectors perform as a surrogate for the emission matrix of an HMM. This model benefits from the property that SVD is globally convergent, and it is shown that it will perform well asymptotically in prediction tasks. However, on finite sample sizes, it can yield negative HMM probability estimates. While there may be *ad hoc* solutions to this concern, imposing a structure constraint on the HMM's transition matrix is not trivial to achieve. Such structure arises from modeling recurring sequential patterns by sub-HMMs, as depicted in Fig. 1.

(ii) Non-negative matrix factorization (NMF) or non-negative matrix tri-factorization (NMTF) to estimate the HMM parameters.

These models aim at learning the emission and transition matrices of an HMM by low-rank non-negative decompositions of the co-occurrence matrix. NMF and NMTF based on Frobenius-norm are discussed in Lakshminarayanan and Raich (2010) and Yang and Oja (2011) respectively. Since Kullback–Leibler divergence (KLD) is an appropriate metric for count data from probabilistic distributions (Fevotte and Cemgil, 2009; Gaussier and Goutte, 2005), it is utilized as the cost function of NMF (Cybenko and Crespi, 2011; Finesso et al., 2010) and NMTF in Vanluyten et al. (2008). The convergence of the algorithms and the experiments on synthetic examples are reported in the articles mentioned above. Promising results are obtained in Cybenko and Crespi (2011) on classical HMM learning problems with a few hidden states. The probabilistic interpretations of NMF (Gaussier and Goutte, 2005) and NMTF (Lakshminarayanan and Raich, 2010) and the ease with which the HMM topology can be imposed on the model are our main motivation to opt for this category of methods as will be presented in Section 3.2.2.

1.2. Pattern discovery by NMF

In earlier work (Van hamme, 2008; Sun and Van hamme, 2011a), we have proposed a different approach to discover recurring patterns in sequential data. In this approach, data sequences as well as patterns are represented as a *bag-of-co-occurrences* (BOC), i.e. unordered frequencies (hence "bag") of ordered symbol pairs (hence "co-occurrences"). A bag-of-co-occurrences is also just a vectorized version of the co-occurrence matrix described in Section 1.1. The BOC representation of a sequence will be the weighted addition of the BOC representations of the recurring patterns. Hence, the pattern BOC representations can be found by NMF from multiple BOC sequence representations. Other linear *latent variable methods* such as SVD could be used to analyze the co-occurrence data. As argued by Lee and Seung (1999), NMF does more than merely reduce dimensionality: it decomposes objects into parts. The objects are the (BOC representations of) utterances, while the parts are the (BOC representations of) patterns.

Unlike the HMM, the NMF model is not a generative model of the sequences themselves but only of their co-occurrence statistics. NMF can thus be viewed as a top-down method that cares about modeling *global*, sequence-level statistics as a sum of parts. Despite the fact that it does not provide as powerful a model as an HMM of local sequentiality of the data, it can successfully discover patterns and subsequently recognize these patterns in unseen data (Sun and Van hamme, 2011a). More recently, other researchers have also constructed latent variable non-HMM acoustic models for speech. Latent Semantic Mapping (Sundaram and Bellegarda, 2010) also uses co-occurrence frequencies of discrete symbols, but unlike (Van hamme, 2008), the decomposition is based on SVD.

1.3. Tensor factorization for unsupervised HMM learning

Assuming that the observation symbols are generated by a HMM of the topology of Fig. 1, i.e. where each pattern is adequately modeled by a left-to-right sub-HMM, the BOC representation of the patterns obtained as in Section 1.2 will show structure. Like the methods discussed in Section 1.1, this structure can be related to the HMM emission and transition probabilities through a NMTF. The effectiveness of the NMF-NMTF learning is witnessed by Sun and Van hamme (2011b) where NMTF successfully found the potential HMM structures from the BOC representations obtained with NMF.

In this article, we opt to arrange the data in a non-negative co-occurrence *tensor*, where each slice of the tensor represents the co-occurrence matrix of a single observation sequence. Each slice will therefore contain the sum of the co-occurrence statistics of the sequential patterns appearing in its observation sequence. Sufficient variation in the occurrence of the sequential patterns in different observation sequences will facilitate the HMM identification.

Like the co-occurrence matrix, the co-occurrence tensor is also expected to show a low-rank structure induced by the HMM. With non-negative Tucker decomposition (NTD) (Mørup et al., 2008; Cichocki et al., 2009), it is possible to approximate the observed co-occurrence tensor by a low-rank tensor. Motivated by the nature of the data (counts), we opt for the KLD between the data and its low-rank approximate model as the cost function to minimize. We establish a relation between the tensor factors and the HMM parameters, leading to a structured tensor factorization problem, which is solved with a two-stage procedure, corresponding to the NMF and NMTF of Section 1.2.

1.4. Outline

With EM and NTD training alternatives, two views on the sequential data are provided: the HMM is good at modeling the *local sequential aspects of the data*, while the NTD does not model the local sequential aspects very well (it only models bigram statistics), but analyses the co-occurrence data into parts corresponding to recurring patterns. Like EM training of HMM, non-negative tensor and matrix decomposition also suffer from local optima (Donoho and Stodden, 2004). The general idea behind the combined approach is that the dual view on the data leads to local optima for which the sub-HMMs correspond better to recurring patterns. This will be shown experimentally in Section 5. Two joint training paradigms of NTD and HMM are proposed to combine the two models. The first one is to apply NTD as a regularizer in EM estimation of the HMM, i.e. using a joint objective function. The second one is alternating training of the NTD and the HMM towards solutions with agreements to both models. The connection between NTD and HMM is bridged by the structured constraints on the tensor factors to ensure that both models share the same group of parameters.

The main contributions of this article are hence a NTD model for data generated by an HMM and methods for jointly learning the NTD and HMM parameters. We show that the proposed method outperforms several baseline EM training methods on speech data.

The outline of the article is as follows: the general description of the HMM is presented in Section 2. The NTD model and its structured factorization are explained in Section 3. The joint training methods are given in Section 4. Experiments of unsupervised vocabulary acquisition and recognition on the TIDIGITS database are reported and discussed in Section 5. We present our conclusions in Section 6.

2. Preliminaries

2.1. HMM configuration

A discrete density HMM for sequential pattern discovery and recognition is configured by the connection of multiple left-to-right sub-HMMs with non-emitting beginning and end states as is shown in Fig. 1. The HMM is characterized by the following elements (Rabiner, 1989).

- The observation alphabet $\mathcal{V} = \{\mathcal{V}_1, \ldots, \mathcal{V}_M\}$ where M is the number of observable symbols.
- The states of the r-th sub-HMM $\{S_{r,1}, \ldots, S_{r,L_r}\}$ where L_r is the number of states in the sub-HMM that models a sequential pattern. For simplicity, L_r is selected as a constant L. $K = RL$ is the total number of hidden states and R is the number sub-HMMs.
- The sequence of hidden states $\mathbf{Q}^{(n)} = \{Q_1^{(n)}, Q_2^{(n)}, \ldots, Q_{T_n}^{(n)}\}$ which are aligned to $\mathbf{O}^{(n)} = \{O_1^{(n)}, O_2^{(n)}, \ldots, O_{T_n}^{(n)}\}$, the n-th training sequence with length T_n. There are N sequences in total, so $1 \leq n \leq N$.
- The emission matrix $\mathbf{B}_{M\times K}$ whose element $B_{m,k}$ is the conditional probability of observation $\mathcal{V}_m$ in state S_k: $\Pr(O_t^{(n)} = \mathcal{V}_m | Q_t^{(n)} = S_k)$, $\forall t$.
- The transition matrix $\mathbf{A}_{K\times K}$ with element $A_{k,k'}$ equal to the conditional probability of transition from S_k to $S_{k'}$: $\Pr(Q_{t+1}^{(n)} = S_{k'} | Q_t^{(n)} = S_k)$, $\forall t$. For the HMM in Fig. 1, $\mathbf{A}$ has a special structure with only non-zero probabilities on the diagonal and sub-diagonal locations
  $\{A_{(r-1)L+l',(r-1)L+l'}, A_{(r-1)L+l,(r-1)L+l+1}\}$ and cross-pattern locations
  $\{A_{rL,(r'-1)L+1}\}$, where $r, r' = 1, \ldots, R$ and $l = 1, \ldots, L-1$ and $l' = 1, \ldots, L$.
- The initial state distribution $\boldsymbol{\pi}$ where $\pi_k = \Pr(Q_1^{(n)} = S_k)$. Only the initial states of the patterns, $\{\pi_{(r-1)L+1}, r = 1, \ldots, R\}$, have non-zero probabilities in $\boldsymbol{\pi}$.

To summarize the above description, the HMM is depicted by a parameter group $\boldsymbol{\Lambda} = \{\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}\}$. All the parameters are non-negative and satisfy $\sum_m B_{m,k} = 1$, $\sum_{k'} A_{k,k'} = 1$ and $\sum_k \pi_k = 1$.

2.2. The EM algorithm for HMM training

The objective function for training an HMM is to maximize the data likelihood in (1),

$$\max_{\boldsymbol{\Lambda}} \quad \sum_{n=1}^{N} \log \Pr(\mathbf{O}^{(n)} | \boldsymbol{\Lambda}). \tag{1}$$

EM algorithm is usually applied to solve (1). Here we briefly introduce the EM algorithm. We will refer to the involved optimization formulations in Section 4.1.

By introducing an auxiliary function (Baum et al., 1970) and using the HMM assumptions to separate the unknown variables $\boldsymbol{\pi}$, $\mathbf{A}$ and $\mathbf{B}$, one obtains the well-known $\mathcal{Q}$-functions as follows:

$$\mathcal{Q}_\pi(\boldsymbol{\pi}, \overline{\boldsymbol{\Lambda}}) := \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_1^{(n)}(k) \log(\pi_k), \tag{2}$$

$$\mathcal{Q}_A(\mathbf{A}, \overline{\mathbf{\Lambda}}) := \sum_{n=1}^{N}\sum_{k,k'=1}^{K}\sum_{t=1}^{T_n-1}\xi_t^{(n)}(k,k')\log(A_{k,k'}), \tag{3}$$

$$\mathcal{Q}_B(\mathbf{B}, \overline{\mathbf{\Lambda}}) := \sum_{n=1}^{N}\sum_{m=1}^{M}\sum_{k=1}^{K}\sum_{t=1}^{T_n}\gamma_t^{(n)}(k)\log(B_{O_t^{(n)},k})\delta_{O_t^{(n)},\mathcal{V}_m}, \tag{4}$$

where $\gamma_t^{(n)}(k) = \Pr(Q_t^{(n)} = S_k | O_t^{(n)}, \overline{\mathbf{\Lambda}})$ is the posterior probability on state S_k of the observation $O_t^{(n)}$ and $\xi_t^{(n)}(k,k') = \Pr(Q_t^{(n)} = S_k, Q_{t+1}^{(n)} = S_{k'} | \mathbf{O}^{(n)}, \overline{\mathbf{\Lambda}})$ is the transition probability from state S_k to state $S_{k'}$. Both of them are estimated by using the current estimate of HMM parameters, $\overline{\mathbf{\Lambda}}$.

The updating algorithms are derived by using Lagrange multipliers:

$$\pi_k = \frac{\sum_{n=1}^{N}\gamma_1^{(n)}(k)}{\sum_{k'=1}^{K}\sum_{n=1}^{N}\gamma_1^{(n)}(k')} \tag{5}$$

$$A_{k,k'} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T_n-1}\xi_t^{(n)}(k,k')}{\sum_{k''=1}^{K}\sum_{n=1}^{N}\sum_{t=1}^{T_n-1}\xi_t^{(n)}(k,k'')} \tag{6}$$

$$B_{m,k} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T_n}\gamma_t^{(n)}(k)\delta_{O_t^{(n)},\mathcal{V}_m}}{\sum_{m'=1}^{M}\sum_{n=1}^{N}\sum_{t=1}^{T_n}\gamma_t^{(n)}(k)\delta_{O_t^{(n)},\mathcal{V}_m}} \tag{7}$$

2.3. HMM and co-occurrence statistics

The relation between the joint distribution of observation variables $\Pr(O_t, O_{t+1})$ and the parameters $\mathbf{\Lambda}$ of the potential HMM can be expressed in Eq. (8) (Hsu et al., 2011; Vanluyten et al., 2008).

$$\begin{aligned} &\Pr(O_t = \mathcal{V}_m, O_{t+1} = \mathcal{V}_{m'}) \\ =\ &\sum_{S_k, S_{k'}} \Pr(O_t = \mathcal{V}_m | Q_t = S_k)\Pr(Q_t = S_k, Q_{t+1} = S_{k'})\Pr(O_{t+1} = \mathcal{V}_{m'} | Q_{t+1} = S_{k'}) \\ =\ &(\mathbf{BTB}')_{m,m'}, \end{aligned} \tag{8}$$

where $\mathbf{T}$ is the joint distribution matrix of hidden states, i.e. $T_{k,k'} = \Pr(Q_t = S_k, Q_{t+1} = S_{k'})$. Provided the prior state probabilities are non-zero, the row normalized $\mathbf{T}$ is equal to the transition matrix $\mathbf{A}$ of the HMM. The joint distribution of observation variables $\Pr(O_t = \mathcal{V}_m, O_{t+1} = \mathcal{V}_{m'})$ can be estimated from the co-occurrence statistics of observation symbols:

$$C_{m,m'} = \frac{1}{T-1}\sum_{t=1}^{T-1}\delta(O_t = \mathcal{V}_m, O_{t+1} = \mathcal{V}_{m'}), \tag{9}$$

where δ is the Kronecker delta and T is the length of the observation sequence. Thus the matrix factorization $\mathbf{C} = \mathbf{BTB}'$ can be used to estimate the HMM parameters $\mathbf{A}$ and $\mathbf{B}$. In previous work on learning HMM by matrix factorization (Lakshminarayanan and Raich, 2010; Yang and Oja, 2011; Cybenko and Crespi, 2011; Finesso et al., 2010; Vanluyten et al., 2008), the co-occurrence matrix $\mathbf{C}$ is obtained by accumulating the co-occurrences of observable symbols over all training sequences. As is explained in the introduction, the resulting statistics originate from potentially all sub-HMMs and unmixing the contributions from the states of each model is a hard task. An ideal solution to tackle this problem could be to learn a sub-HMM for each pattern separately where each sub-HMM has a relatively small number of hidden states. However, we do not know which patterns the data includes, nor the segmentation boundaries between patterns. To get around this difficulty, blind source separation techniques are applied to discover the latent components corresponding to patterns.

3. Non-negative Tucker decomposition for HMM training

In this section, the contributions of the sub-HMMs to the co-occurrence statistics are separated using NMF applied to a data tensor. Subsequently, NMTF is applied to estimate the HMM parameters.

3.1. Non-negative tensor representation and decomposition

The relation between the NTD and the HMM is now explored which will lead to structured constrained decomposition. The reader is referred to Appendix A for notations and elementary tensor operations. Co-occurrence statistics of all training sequences are represented in a M-by-M-by-N tensor $\underline{\mathbf{O}}$. Its mode-3 slice, $\underline{\mathbf{O}}_{(3)}^{(n)}$, contains the normalized co-occurrence matrix of sequence n with elements:

$$\underline{O}_{m,m',n} = \frac{1}{T_n - 1} \sum_{t=1}^{T_n-1} \delta(O_t^{(n)} = \mathcal{V}_m, O_{t+1}^{(n)} = \mathcal{V}_{m'}). \tag{10}$$

Consider a low-rank decomposition of $\underline{\mathbf{O}}$ into a core tensor $\underline{\mathbf{T}}$ (Tucker, 1966) and the coefficient matrices $\mathbf{B}$ and $\mathbf{H}$ in Eq. (11) and in (a3) of Fig. A.8 (Mørup et al., 2008; Cichocki et al., 2009).

$$\underline{\mathbf{O}} \approx \underline{\mathbf{T}} \times_1 \mathbf{B} \times_2 \mathbf{B} \times_3 \mathbf{H}', \tag{11}$$

- $\mathbf{B}$ is the emission matrix of the HMM where its columns from $(r-1)L+1$ to rL correspond to the HMM states of the sub-HMM r. The mode-1 and mode-2 factorization in Eq. (11) is symmetric with respect to $\mathbf{B}$ and follows from Eq. (8).
- Orthogonality among the mode-3 slices of $\underline{\mathbf{T}}$ is imposed. The r-th mode-3 slice of $\underline{\mathbf{T}}$, $\underline{\mathbf{T}}_{(3)}^{(r)}$, is the *extended* joint distribution matrix of the states of sub-HMM r, $\mathbf{T}^{(r)}$. As shown at the bottom of Fig. 2, $\underline{\mathbf{T}}_{(3)}^{(r)}$ is constructed from $\mathbf{T}^{(r)}$ as $\underline{\mathbf{T}}_{(3)}^{(r)} = \text{blkdiag}(0, \ldots, 0, \mathbf{T}^{(r)}, 0, \ldots, 0)$ such that its size is equal to that of the transition matrix $\mathbf{A}$ of the HMM. Moreover, $\mathbf{T}^{(r)}$ has a sub-diagonal structure for the HMM configuration in Fig. 1. The transitions between the sub-HMMs will be explained in Section 4.
- $\mathbf{H}$ is the weight matrix of the sub-HMMs in the sequences, i.e. $H_{r,n}$ is the weight of sub-HMM r in sequence n. In the case where a single sub-HMM refers to a single whole-word, the activation $H_{r,n}$ is the probability of occurrence of word r in the utterance n, similar to the occurrence probability of a topic in a document in the area of document retrieval.

The NTD problem is split into an NMF problem followed by R NMTF problems. The NMF is motivated as follows: $\underline{O}_{m,m',n}$ is an estimate of $\Pr((\mathcal{V}_m, \mathcal{V}_{m'})|\text{utt}_n)$ which can be factorized as:

$$\Pr((\mathcal{V}_m, \mathcal{V}_{m'})|\text{utt}_n) \approx \sum_r \Pr((\mathcal{V}_m, \mathcal{V}_{m'})|\text{sub-HMM}_r)\Pr(\text{sub-HMM}_r|\text{utt}_n). \tag{12}$$

Hence we can first reshape $\underline{\mathbf{O}}$ into a matrix and apply matrix factorization to unmix the co-occurrence statistics into sub-HMM-specific co-occurrence statistics. Each mode-3 slice of the data tensor $\underline{\mathbf{O}}$, $\underline{\mathbf{O}}_{(3)}^{(n)}$, thus characterizes an observation sequence by its symbol co-occurrence counts. Inspired by the terminology used in the area of document retrieval, we call this a *bag-of-co-occurrence* representation, emphasizing that the order in which the co-occurrence events appeared in the data is disregarded. Subsequently, the sub-HMM-specific co-occurrences are decomposed into matrices that relate to the HMM parameters.

A two-stage NMF-NMTF algorithm is applied to realize the above procedures as is shown in Eq. (13) and Fig. 2.

$$\begin{aligned} &\text{NMF}: && \underline{\mathbf{O}} \approx \underline{\mathbf{C}} \times_3 \mathbf{H}', \\ &\text{NMTF}: && \underline{\mathbf{C}}_{(3)}^{(r)} \approx \mathbf{B}^{(r)}\mathbf{T}^{(r)}(\mathbf{B}^{(r)})', \\ &\text{Stacking}: && \mathbf{B} = [\mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(R)}], \ \underline{\mathbf{T}}_{(3)}^{(r)} = \text{blkdiag}(0, \ldots, 0, \mathbf{T}^{(r)}, 0, \ldots, 0), \end{aligned} \tag{13}$$

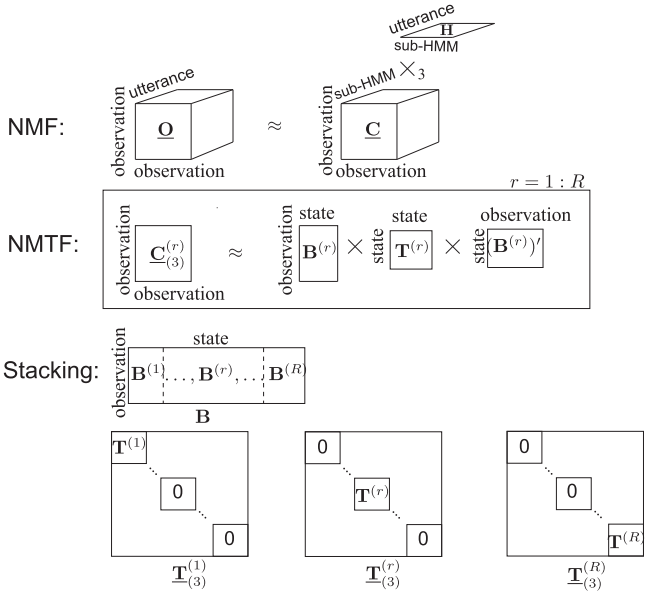

Fig. 2. Structured non-negative Tucker decomposition $\underline{\mathbf{O}} \approx \underline{\mathbf{T}} \times_1 \mathbf{B} \times_2 \mathbf{B} \times_3 \mathbf{H}'$ in three stages: (top) NMF to extract recurrent sequential patterns $\{\underline{\mathbf{C}}^{(r)}_{(3)}, r = 1, \ldots, R\}$, (middle) NMTF is applied subsequently to learn a sub-HMM for each pattern r, (bottom) stacking the R emission matrices in one emission matrix $\mathbf{B}$ and expanding the R transition matrices with zeros and stacking these in the core tensor as mode-3 slices $\mathbf{T}^{(r)}_{(3)}$.

The size of the tensor $\underline{\mathbf{O}}$ is $M \times M \times N$. By reshaping the mode-3 slices of $\underline{\mathbf{O}}$ into vectors, NMF is first utilized to extract R recurring patterns represented by co-occurrences of observation symbols in a low-rank tensor $\underline{\mathbf{C}}$ with size $M \times M \times R$. The weight matrix $\mathbf{H}$ thus has size $R \times N$. For each mode-3 slice $\underline{\mathbf{C}}^{(r)}_{(3)}$ which is a $M \times M$ matrix, NMTF is subsequently applied to find the $M \times L$ emission matrix $\mathbf{B}^{(r)}$ and the $L \times L$ transition matrix of the corresponding sub-HMM. Finally the emission matrix $\mathbf{B}$ of the HMM is obtained by stacking the emission matrices of the sub-HMMs. The core tensor $\underline{\mathbf{T}}$ is constructed by extending and stacking the $\mathbf{T}^{(r)}$'s from the sub-HMMs.

3.2. Decomposition algorithms

The algorithms to solve the NMF and NMTF problems of Eq. (13) are described in this subsection.

3.2.1. NMF with column-wise sum-to-unity constraints

The algorithm used for computing the NMF can be found in Lee and Seung (2000). Adopting the notations of Lee and Seung (2000), the problem is,

$$\text{argmin}_{\mathbf{W},\mathbf{H}} \text{KLD}(\mathbf{V}||\mathbf{W}, \mathbf{H}), \tag{14}$$

where $\text{KLD}(\mathbf{V}||\mathbf{W},\mathbf{H}) := \sum_{i,n} V_{i,n} \log(V_{i,n}/(WH)_{i,n}) - V_{i,n} + (WH)_{i,n}$. The matrix $\mathbf{V}$ to be factorized is reshaped from the input tensor $\underline{\mathbf{O}}$ by $\mathbf{V} = [\text{vec}(\underline{\mathbf{O}}^{(1)}_{(3)}), \ldots, \text{vec}(\underline{\mathbf{O}}^{(N)}_{(3)})]$ where *vec* is the operator to transform a matrix to a vector by stacking its columns.

With the factorization, one obtains $\mathbf{W}$ and $\mathbf{H}$ where the relation between $\mathbf{W}$ and the tensor $\underline{\mathbf{C}}$ is $\mathbf{W} = [\text{vec}(\underline{\mathbf{C}}^{(1)}_{(3)}), \ldots, \text{vec}(\underline{\mathbf{C}}^{(R)}_{(3)})]$. $\mathbf{H}$ is the weight matrix that holds the activation probabilities of the sub-HMMs both in Lee and Seung (2000) and in this article. The NMF achieves a dimension reduction by finding repeated parts $\mathbf{W}$ from the original data $\mathbf{V}$. The columns of $\mathbf{W}$ have been demonstrated to be strongly related to vocabularies in Sun and Van hamme (2011a), each of which is represented in its bag-of-co-occurrences format. In order to make the model consistent with the probabilistic latent semantic analysis (Hoffman, 1999) framework, the columns of $\mathbf{V}$ and $\mathbf{W}$ are normalized to unity. Later in Section 5.2.3 we will demonstrate that the NMF and HMM can benefit from each other during the joint training.

Table 1
Non-negative matrix tri-factorization algorithm minimizing KLD($\mathbf{Z}\|\mathbf{XYX}'$).

Input:	Data matrix $\mathbf{Z}$ and initial guess $\mathbf{X}$,$\mathbf{Y}$
While Stopping criterion is not met	
	$\mathbf{F} = \mathbf{Z} \oslash (\mathbf{X} * \mathbf{Y} * \mathbf{X}')$;
	$\mathbf{X} \leftarrow \mathbf{X} \odot (\mathbf{F} * \mathbf{X} * \mathbf{Y}' + \mathbf{F}' * \mathbf{X} * \mathbf{Y})$;
	$X_{i,k} \leftarrow X_{i,k} / \sum_{i'} X_{i'k}$;
	$\mathbf{Y} \leftarrow \mathbf{Y} \odot (\mathbf{Y}' * (\mathbf{Z} \oslash (\mathbf{X} * \mathbf{Y} * \mathbf{X}')) * \mathbf{Y})$;
End	
Output:	$\mathbf{X}$, $\mathbf{Y}$

3.2.2. NMTF with sum-to-unity constraints

The NMTF algorithm for computing the second line in Eq. (13) is shown in Table 1, while its derivation and proof of convergence are given in Appendix B. The NMTF is conducted for the co-occurrence matrix of each pattern discovered by NMF, i.e. the input is $\mathbf{Z} = \underline{\mathbf{C}}^{(r)}_{(3)}$, $1 \le r \le R$. The outputs are $\mathbf{T}^{(r)} = \mathbf{Y}$ and $\mathbf{B}^{(r)} = \mathbf{X}$.

The structure constraints on $\mathbf{T}^{(r)}$ are achieved by proper initialization of $\mathbf{Y}$ to a sub-diagonal structure. With the zero-locking property of the multiplicative update (Lee and Seung, 2000), $\mathbf{Y}$ will keep this structure during subsequent iterations. Each column of $\mathbf{X}$ is normalized to unity column-wise to ensure it is an emission matrix of the sub-HMM. An interesting property is $\sum_{k,k'} Y_{k,k'} = \sum_{m,m'} Z_{m,m'} = 1$ given $\sum_m X_{m,k} = 1$, so $\mathbf{Y}$ can be interpreted as a joint probability.

Like in other work (Cichocki et al., 2009; Berry et al., 2006), the stopping criterion for both NMF and NMTF can be a fixed number of iterations or a sufficiently small change of the objective function.

4. Joint training of NTD and HMM

The state posterior probability computation in the EM algorithm is a soft assignment of frames to states. State posteriors are computed per utterance, i.e. form a *local view* on the data. The local view on the data is aggregated in a global HMM parameter re-estimation by accumulating the observations weighted by the state posteriors. By contrast, NTD does not perform local alignments but unmixes the co-occurrence statistics into sub-HMM-specific co-occurrence statistics using model (12) (which implies the co-occurrence events are independent given the sub-HMM) and subsequently estimates the HMM parameters using NMTF, a model that is motivated by the HMM independence assumptions. The optimization problems in NTD are not convex, so they could also be trapped in poor local optima. The bag-of-co-occurrence representation of a sequence in NMF of Eq. (13) also loses sequential information which could result in models which contradict the sequential order. In other words, while the function of the NMF is to unmix the co-occurrence statistics originating from the different sequential patterns, there is no constraint on $\underline{\mathbf{C}}$ to express that the bag-of-co-occurrence representation of the patterns contained in its columns can actually be generated from an observation sequence. Below, this constraint is achieved by linking the NTD the Baum–Welch (BW) algorithm.

A transformation φ and its inverse between the core tensor $\underline{\mathbf{T}}$ and the transition matrix $\mathbf{A}$ of the HMM are first defined as follows.

$$\begin{aligned} \varphi: \quad & \mathbf{A} = \sum_r \underline{\mathbf{T}}^{(r)}_{(3)} + \mathcal{E}, \\ & A_{k,k'} \leftarrow A_{k,k'} / \sum_{k'} A_{k,k'}, \\ \varphi^{-1}: \quad & \mathbf{T}^{(r)} = \mathbf{A}_{(r-1)*L+1:r*L,(r-1)*L+1:r*L}, \\ & \underline{\mathbf{T}}^{(r)}_{(3)} = \mathrm{blkdiag}(0, \ldots, 0, \mathbf{T}^{(r)}, 0, \ldots, 0). \end{aligned} \tag{15}$$

$\mathcal{E}$ is a matrix to connect the end state of one pattern and the beginning state of any other pattern. It is initialized with a positive constant ϵ with scale $1/K$ where K is the total number of states, that is $\mathcal{E}_{rL,(r'-1)L+1} = \epsilon$ with $r = 1, \ldots, R$ and zero elsewhere. $\mathcal{E}$ will be updated during the BW iterations.

4.1. NTD regularized BW (NTD.Reg.BW)

The first way to combine NTD and HMM is to construct a joint objective function by treating NTD as a regularizer as in Eq. (16).

$$\max_{\mathbf{A},\mathbf{B},\boldsymbol{\pi},\mathbf{H}} \sum_n \log\Pr(\mathbf{O}^{(n)};\mathbf{A},\mathbf{B},\boldsymbol{\pi}) - \lambda * \mathrm{KLD}(\underline{\mathbf{O}}||\varphi^{-1}(\mathbf{A}) \times_1 \mathbf{B} \times_2 \mathbf{B} \times_3 \mathbf{H}'). \tag{16}$$

The first term is the log-likelihood of the observed sequences $\{\mathbf{O}^{(n)}\}_{n=1}^N$ on its HMM, and the second term is the Kullback–Leibler divergence between the data tensor $\underline{\mathbf{O}}$ and its reconstruction. The HMM parameter estimates will fit both data representations: the first term measures how well the HMM is able to generate the actual sequences, while the regularization term expresses that the global co-occurrence view can be decomposed into additive parts along the lines of Section 1.2. $\mathbf{H}$ only occurs in the second term, so its update algorithm remains unchanged from Section 3. Similarly, $\boldsymbol{\pi}$ only occurs in the HMM so EM updates can be used. The optimization of $\mathbf{A}$ and $\mathbf{B}$ in the HMM and the NTD is based on auxiliary functions which boil down to the following optimization problem:

$$\begin{aligned} \max_{y_1,\ldots,y_M} \quad & \mathcal{L}(y_1,\ldots,y_M) = \sum_{m=1}^{M} \mu_m \log(y_m) + \lambda\omega_m \log(y_m), \\ \text{s.t.} \quad & \sum_{m=1}^{M} y_m = 1,\ y_m \geq 0. \end{aligned} \tag{17}$$

where $\mathbf{y} = (y_1, \ldots, y_M)$ denotes a column of $\mathbf{B}$ or a row of $\mathbf{A}$, μ_m is the term of constant (w.r.t. $\mathbf{y}$) derived from the auxiliary function of the EM algorithm of an HMM (Eqs. (3) and (4)) and ω_m is the term derived from the auxiliary function of the NTD (Eq. (27) in Lee and Seung (2000) and Eq. (B.3) in this article). Hence, the joint update algorithm obtained by applying Lagrange multipliers to Eq. (17) is as follows.

$$y_m = \frac{\mu_m + \lambda\omega_m}{\sum_{m=1}^{M} \mu_m + \lambda\omega_m} \tag{18}$$

In Eq. (18), the μ_m sum to approximately the total sequence length, while the ω_m sum to about 1. Hence, a regularization constant in the order of magnitude of the total sequence length should be expected in order for the regularization term to play a role.

With the optimization algorithm, the joint objective function in Eq. (16) will increase monotonically. However, in the above procedures, the iterations in NTD and HMM are *synchronous*, i.e. both of them are updated with the same number of iterations. This may not be the best choice according to our experience where NTD normally requires much more iterations to convergence (e.g. hundreds or thousands) than the iterations in the EM training of an HMM. So though this method ensures non-decrease of the joint objective function, the synchronous update may be too slow for practical purposes.

4.2. Alternating training of NTD and BW (NTD.Alt.BW)

A method to combine NTD and BW in an asynchronous manner is to plug the NTD into the EM iterations of BW as is illustrated in Fig. 3 where NTD and BW switch at every α EM training iterations.

The inputs of the training scheme are the observation sequence group $\{\mathbf{O}^{(n)}\}_{n=1}^N$, the non-negative tensor $\underline{\mathbf{O}}$ and the initializations $\boldsymbol{\Lambda} = \{\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}\}$. $\boldsymbol{\pi}$ is initialized to an uniform distribution over all initial states and is set to zero for all other states. $\mathbf{B}$ is initialized to *i.i.d* random variables with an uniform distribution between zero and one and its columns are then normalized to suffice the sum-to-one constraint. $\mathbf{A}$ is initialized as a transition matrix of the HMM in Fig. 1. The loop and transition probabilities in $\mathbf{A}$ are first generated from an uniform distribution between zero and one. The self-loop and outgoing probabilities of each state are subsequently normalized to sum to unity.

The initial HMM parameter estimates $\boldsymbol{\Lambda} = \{\boldsymbol{\pi}, \mathbf{A}, \mathbf{B}\}$ are transformed to the initial estimates of the NTD parameters: the matrix $\mathbf{B}$ and the core tensor $\underline{\mathbf{T}}$. Subsequently NTD as presented in Section 3 is performed. The updated $\underline{\mathbf{T}}$ is transformed back to its state transition form $\mathbf{A}$ and together with the through NTD updated emission matrix $\mathbf{B}$, several

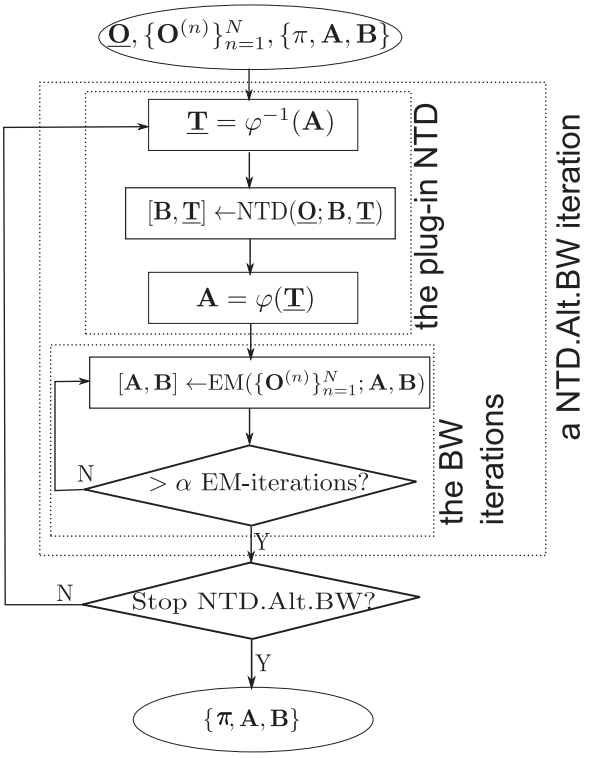

Fig. 3. The alternating training framework of NTD.Alt.BW. The current HMM parameters are transformed to their tensor form by φ^{-1} to initialize the structured tensor decomposition. Then the tensor parameters are transformed back to HMM parameters by φ and α BW iterations are performed. This process is repeated several times.

EM iterations are performed. The updated $\{\mathbf{A}, \mathbf{B}\}$ are used to initialize the NTD. The process is iterated until the stopping criterion is met.

The NTD block in Fig. 3 is computed as presented in Section 3. The block of EM training is approached by the Baum–Welch training (Rabiner, 1989; Baum et al., 1970). The stopping criterion of the EM training and the overall NTD.Alt.BW training is a sufficient number of iterations over the training data or an insufficient increase of the data likelihood. Much like is common practice in EM training of HMMs for speech processing, we have opted for a fixed number of EM iterations α and a fixed number of NTD.Alt.BW iterations β. The total number of EM iterations is thus $\alpha \times \beta$, a parameter that will be studied in Section 5.

5. Experiments on unsupervised spoken pattern discovery and recognition

The proposed methods for unsupervised learning of HMMs are evaluated on a spoken pattern discovery task from a set of utterances, each composed of a set of recurring words. With the HMM configuration in Fig. 1, each sub-HMM is expected to model a spoken word. However, the number of words is not known beforehand. It is assumed that some knowledge about the number of words is available, though the exact number is not critical. Since it is conceivable that multiple sub-HMMs model a single pattern, the actual number of sub-HMMs, R, is chosen greater than the expected number of words.

5.1. Data and experiments

The experiments are carried out on the TIDIGITS database which has a vocabulary of 11 English digits in 8438 training utterances and 1001 test utterances in continuous speech from multiple male and female speakers (Hirsch and Pearce, 2000). The advantage of this database is that it is easy to analyze what the method discovers.

5.1.1. Gaussian identities as observation symbols

Like in conventional ASR systems, speech is chopped into overlapping frames to compute the short-term spectral features known as Mel-Frequency Cepstral Coefficients (MFCC) (Huang et al., 2001). The window length is 25 ms with a frame shift of 10 ms. For each frame, a 12 dimensional MFCC is extracted from a bank of 30 Mel-scaled filters. Each frame is represented by its MFCC plus the frame's log-energy. First and second order differences are computed and concatenated to form a 39-dimensional feature vector. A Gaussian mixture of $M = 1000$ components is trained without supervision on the training set using maximum likelihood EM training. The Gaussian identities will later be utilized as discrete observable symbols.

5.1.2. Representation of the training data

A training utterance, indexed by n, is characterized by two representations: an observation sequence $\mathbf{O}^{(n)}$ and a co-occurrence matrix $\underline{\mathbf{O}}^{(n)}_{(3)}$. The sequence is obtained by quantizing each frame as the Gaussian identity with the highest posterior probability. The co-occurrence matrix is computed from the accumulated co-occurrences of Gaussian symbols of this utterance, as is presented in Section 3. The complete training data is hence described as a set of sequences $\{\mathbf{O}^{(n)}\}_{n=1}^{N}$ and a non-negative tensor $\underline{\mathbf{O}}$ with count data.

5.1.3. Parameter settings

In the first experiments, we choose $R = 30$ sub-HMMs, each with $L = 10$ states. For training the NTD regularized BW (NTD.Reg.BW), the regularization parameter is selected to be $\lambda = 10^4$ (e.g. in the order of magnitude of the total sequence length) and the number of EM iterations is 25. In the alternating training of NTD and BW (NTD.Alt.BW), the Baum–Welch iteration is initialized from an NTD and is then interrupted by the plug-in NTD every $\alpha = 5$ iterations where the NTD uses 100 iterations. The total number of these NTD interruptions is $\beta = 5$. Hence the total number of EM iterations in HMM is also 25 and the total number of iterations in NTD is 500. Note that NTD iterations require far less computational resources than EM iterations because there is no need to process every data frame, but only the co-occurrence statistics which are computed once.

Since we are dealing with non-convex optimization, initialization is important. We generate five groups of initial HMM parameter values randomly, which are each used as a starting point for all methods. For each method, we thus obtain five performance metrics of which the variance is observed, which constitutes an analysis of the sensitivity of the methods initialization.

5.2. Evaluation and discussion

5.2.1. Baselines

Three methods for unsupervised learning of HMMs are experimented with as baselines:

- **BW** is Baum–Welch training (Roweis, 1999), which estimates the HMM parameters such that the data likelihood is maximized given the HMM topology of Fig. 1. Like in the proposed methods, 25 iterations are performed starting from the same HMM.
- **BW+SA** is BW training with simulated annealing (SA) (Paul, 1985), in which the HMM parameter estimates are randomly perturbed by an amount that is proportional to $\exp(-0.1i)$ where i is the iteration number. Because of the perturbations, the number of iterations was set to 100 here. At the end of the cooling, an additional 25 perturbation-free iterations are conducted to ensure the final convergence.
- **JLH** is Joint Label and HMM parameter estimation as used in Siu et al. (2011) to construct so-called self-organized units to represent speech. In summary, this method alternates between estimating the pattern labels given the HMM (recognition) and estimating the HMM given the labels (supervised training). While it is relevant to apply this optimization method to word discovery, the initialization used in Siu et al. (2011) is not: it starts form a spectral discontinuity metric to segment the audio into phone-like segments, which would make it hard for JLH to converge to word-sized units. Therefore, JLH was initialized with the output of the BW training or with the output of NTD.Alt.BW, after which 25 iterations are performed.

Table 2
Evaluation of four methods for unsupervised learning of discrete density HMMs ($R = 30$, $L = 10$): log-likelihood of the training data, segmentation accuracy in %, model purity in % and recognition accuracy in %. The bottom row shows the corresponding results from supervised training.

	Log-likelihood ($\times 10^6$)	Segmentation	Purity	Recognition
BW	-5.62 ± 0.04	61.6 ± 3.0	76.1 ± 2.6	62.2 ± 5.8
BW+SA	-5.49 ± 0.05	65.0 ± 1.1	79.6 ± 3.5	69.6 ± 5.3
NTD.Reg.BW ($\lambda = 10^4$)	-5.71 ± 0.07	63.9 ± 1.2	77.3 ± 3.4	70.9 ± 5.1
NTD.Alt.BW	-5.74 ± 0.05	78.6 ± 1.5	81.0 ± 2.0	85.7 ± 1.4
NTD.Alt.BW+SA	-5.53 ± 0.02	74.3 ± 3.3	88.6 ± 1.5	85.0 ± 3.7
Oracle	-14.82	91.0	93.3	97.4

All three baseline methods are initialized from the same five random HMM parameter groups described in Section 5.1.3. To show how close to supervised training the unsupervised methods come, a HMM with 12 sub-HMMs (11 for digits with 10 states each and 1 for silence with 3 states) is trained with 25 Viterbi iterations by using manually labeled transcriptions as supervision. The corresponding results are shown in the bottom row ("Oracle") of Table 2.

5.2.2. *Evaluation metrics*

The models are evaluated by the following criteria.

(1) Data likelihood

We report the final log likelihoods $\sum_n \log \Pr(\mathbf{O}^{(n)}; \mathbf{A})$ for each of the methods. Monotonous increase of the likelihood was observed in all the models except BW+SA and NTD.Alt.BW. Since NTD maximizes a different criterion, the data likelihood on the HMM decreases, but subsequent EM iterations quickly recover as is illustrated in the top panel of Fig. 4. The middle panel shows that the NTD interruptions are actually beneficial to the accuracy (as defined below) compared to only one initializing NTD prior to the EM iterations (NTD.Init.BW). The benefit is also reflected in the final log likelihood, which are -5.707 and -5.711 ($\times 10^6$) for NTD.Alt.BW and NTD.Init.BW respectively. In the experiments on TIDIGITS, all the training schemes were observed to converge with final likelihoods listed in the first column of Table 2.

BW+SA successfully found solutions with high data likelihood, but note that due to the nature of the method, 125 EM iterations were run here while 25 iterations were performed for the other methods. Comparing the likelihoods with the other evaluation metrics below, we experimentally conclude that data likelihood may not be a perfect choice for measuring the quality of unsupervised learning methods.

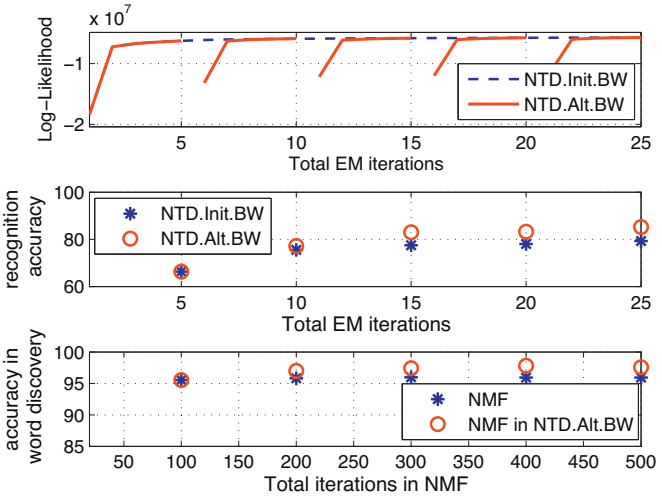

Fig. 4. Convergence analysis. Top: log-likelihood of the training data for NTD.Alt.BW and initializing BW with only one NTD (NTD.Init.BW). Middle: recognition accuracy in % of the intermediate HMMs. Bottom: accuracy in % of the intermediate NMF models.

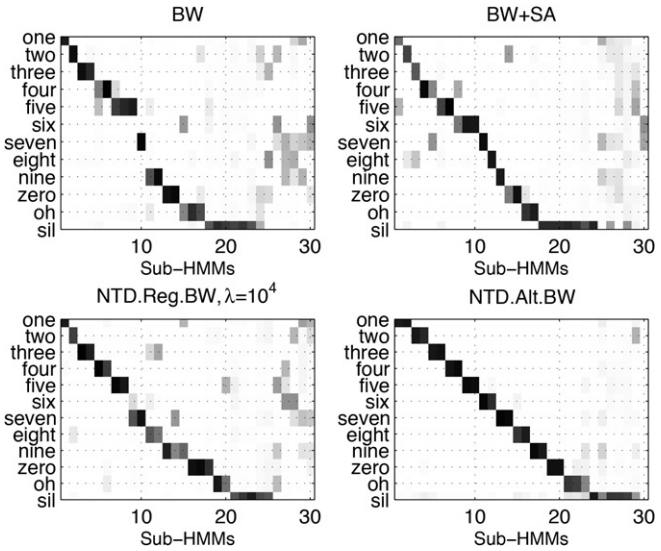

Fig. 5. Mapping matrices between the sub-HMMs and the digits for four different methods initialized equally ($R = 30$, $L = 10$).

(2) Speech segmentation

A second evaluation criterion is segmentation accuracy. The reference segmentation is obtained by segmenting the training utterances with a HMM that was trained on the same data but using the orthographic transcriptions as supervisory information. The accuracy metric used here compares the reference segmentation boundaries b_i, $1 \leq i \leq I$ of an utterance with its boundaries $\tilde{b}_j$, $1 \leq j \leq J$ obtained with the unsupervised methods through a dynamic programming alignment of both boundary lists. First, all boundary pairs are compared with a threshold ξ to generate a distance matrix $d_{i,j}$: their distance is zero if below or equal to the threshold and one otherwise. Then the path of lowest cost from $d_{1,1}$ to $d_{I,J}$ is sought allowing only diagonal, horizontal or vertical transitions. The total cost is summed over all utterances and divided by the total number of boundaries in the reference segmentations of the whole training corpus. The segmentation accuracy is subsequently computed by subtracting the above cost value from 100.

The results with the 5 random initializations with $\xi = 5$ frames (50 ms) are shown in Table 2. The discovered sequential patterns in all the models have strong relations with the digits, but the patterns of NTD.Alt.BW are better than those trained only by BW (first row). Simulated annealing (second row) and NTD.Reg.BW (third row) improve slightly but consistently over BW.

(3) Average purity of the discovered patterns

Because the sequential patterns have been learned without supervision, they are anonymous, i.e. not identified as one of the words in the vocabulary. In order to analyze the discovered patterns, a mapping matrix $\mathbf{G}$ between the digits and the sub-HMMs is first constructed. Hereto, $\mathbf{G}$ is constructed as the co-occurrence matrix between ground truth digits and recognized sub-HMMs. The ground truth is approximated by a segmentation generated with a supervised HMM, yielding a digit label for every data frame. $G_{j,r}$ is then the number of frames that are labeled as digit j ($1 \leq j \leq J$) in the ground truth and as sub-HMM r ($1 \leq r \leq R$) at the end of the unsupervised training. $J = 12$ since we also define a silence word.

Some examples of mapping matrices obtained with different methods using the same initialization are shown in Fig. 5. Here the columns (sub-HMMs) have been sorted according to the digits they represent. Most sub-HMMs are strongly related to one digit, while some are mixtures of different digits (e.g. pattern 24–30 in the top left panel). Some digits are modeled by multiple patterns (e.g. digit 5 in all the panels), which is not undesired as this constitutes a model of variability that is not related to semantics, such as gender or pronunciation variation. Pattern 30 in the top panels is probably a sub-word unit of "s" because the pattern is strongly related to both "six" and "seven" which share the same phone "s". As can be seen in Fig. 5, the NTD.Alt.BW training methods show good performance to extract pure patterns.

To make the visual evaluation of Fig. 5 more quantitative, the purity of sub-HMM r is defined as $\max_{l'} G_{l',r} / \sum_{l'} G_{l',r}$. The best achievable purity is 100%, where each sub-HMM is strictly mapped to one digit or silence. The results are

Table 3
Evaluation of the JLH method: 25 JLH iterations, initialized from the BW and NTD.Alt.BW models respectively ($R = 30$, $L = 10$).

	BW	JLH init. by BW	NTD.Alt.BW	JLH init. by NTD.Alt.BW
Log-likelihood ($\times 10^6$)	-5.56 ± 0.03	-5.67 ± 0.03	-5.58 ± 0.02	-5.77 ± 0.04
Segmentation	61.8 ± 3.0	61.1 ± 2.7	79.7 ± 2.8	77.7 ± 2.0
Purity	77.1 ± 2.3	76.3 ± 2.4	85.4 ± 1.5	83.8 ± 1.3
Recognition	61.3 ± 3.7	51.6 ± 7.4	87.4 ± 3.8	89.0 ± 3.0

reported in Table 2. The NTD.Alt.BW training outperforms the other methods in finding patterns with high purity. The models obtained with simulated annealing also score well in terms of purity. Combining NTD.Alt.BW and simulated annealing improves the purity further to 88.6%.

(4) Recognition accuracy on the test set

In this section, the learned sub-HMMs are used for recognition using Viterbi decoding. In order to be able to score the recognition result, each sub-HMM must be associated to a digit or silence. We rely on the mapping matrices $\mathbf{G}$ obtained above to achieve this. More sophisticated methods such as inducing a finite state grammar from a lattice recognition output (Mohri et al., 2002) could be proposed. To implement the co-occurrence method, we first normalize the columns of the mapping matrix $G_{j,r} \leftarrow G_{j,r}/\sum_{j'} G_{j',r}$ and then relate each sub-HMM to either a digit or garbage (i.e. silence, mixture or sub-word that does not relate clearly to any digit). A threshold η is utilized to identify the garbage patterns $\{r|\max_{l'} G_{l',r} < \eta\}$, for example pattern 24–30 in the top left panel of Fig. 5. If some digit is not modeled by any sub-HMM, we find the garbage model with maximal joint probability with the digit. In this way, in the top left panel of Fig. 5, digit "six" is modeled by pattern 30 and digit "eight" is modeled by pattern 25.

The learned HMM is subsequently applied as a word recognizer on the test set. For each method, the word entry penalty corresponding to the minimal error rate on a fixed subset of 50 utterances from the training set is selected. Since all methods benefit from the same tuning advantage, none is favored over the other. The results are shown in the rightmost column of Table 2 for $\eta = 0.6$. The accuracy is computed by subtracting the word error rate (including insertions, deletions and substitution) from 100. Occurrences of silence and garbage in the recognized string are removed for scoring. The NTD.Alt.BW training improves the baselines significantly. BW+SA and NTD.Reg.BW have also improved over BW in this respect. For comparison, a HMM trained with supervision with 25 Viterbi iterations yields an accuracy of 97.4%.

5.2.3. Analysis and discussion

(1) Baselines

For the baseline of *JLH with different initializations*, Table 3 summarizes the impact of performing JLH iterations initialized from the BW or NTD.Alt.BW models, as explained in Section 5.2.1. The performance of JLH relies heavily on its initialization and is unable to consistently improve over its initialization. Note that in this table, the results of BW and NTD.Alt.BW are obtained by running 25 extra iterations on the ones in Table 2 so the total numbers of EM iterations of all the four methods in Table 3 are the same.

The method *BW+SA* performs better than BW, especially in recognition accuracy. However, SA requires more training time. Because of the random perturbations, results are more difficult to reproduce than for other methods. BW+SA is successful in finding a solution that has better likelihood on the training data. However, it's sub-HMMs are not as pure as what is obtained with NTD.Alt.BW as shown in Table 2 and Fig. 6. Closer inspection of segmentation results shows that the sub-HMMs found by BW+SA contain both phone-like sub-words and larger word-sized units that are not pure. In the top right panel of Fig. 5 for instance, four sub-HMMs are mapped to the digit "six". Upon inspection of segmentations, we observe they often occur in the same order. The first sub-HMM (the 8-th column), however, is also observed as the first sub-HMM of "four", i.e. it could be modeling the phones "s" and "f". Inspection of segmentations also shows that the 15-th sub-HMM is modeling both "two" and "zero" as a whole-word model. For NTD.Alt.BW such phenomena appear not as often as is witnessed by Fig. 5 and the high values of segmentation accuracy, purity and recognition accuracy.

(2) The role of NTD in HMM training

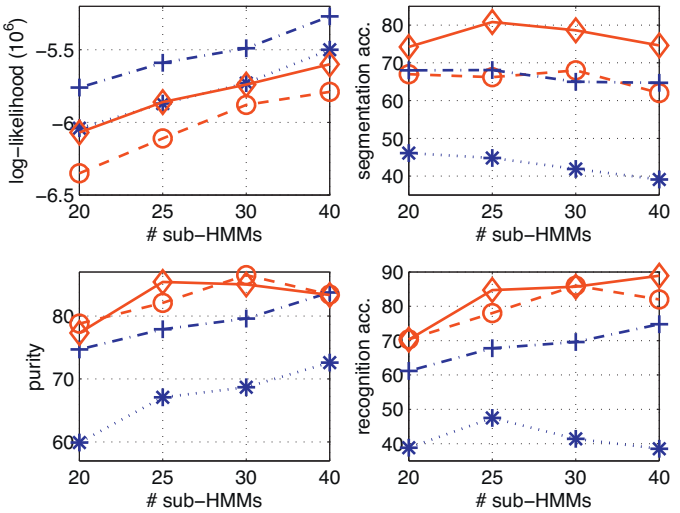

Fig. 6. The performance of BW+SA and NTD.Alt.BW with respect to numbers of states and number of sub-HMMs. ∗: BW+SA with $L = 5$ states, ○: NTD.Alt.BW with $L = 5$ states, +: BW+SA with $L = 10$ states and ◊: NTD.Alt.BW with $L = 10$ states.

Consistent improvement is observed from BW to NTD.Reg.BW for all five initializations and for the performance metrics of segmentation, purity and recognition, though the gain is not as large as for NTD.Alt.BW. NTD.Reg.BW has a well-formulated optimization framework, but performs worse than NTD.Alt.BW in terms of segmentation accuracy, purity and recognition accuracy. The reason is attributed to the synchronous updating of NTD and BW. From past experience, we know that NTD requires hundreds of iterations to converge. In NTD.Reg.BW the KLD between the data tensor and its approximate factorization modifies the EM update rules. Since we compare all methods on the same number of EM iterations, there is a serious concern that the tensor factorization is insufficiently converged in NTD.Reg.BW. More iterations do lead to slight performance improvement, but NTD.Reg.BW never reaches the performance of NTD.Alt.BW.

The middle panel in Fig. 4 shows how the recognition accuracy of the HMMs evolves through the iteration process. The intermediate HMMs in NTD.Alt.BW perform better than the BW training with the same initializations and with the same number of iterations. Similarly, the quality of the NMF models in Eq. (13) during the iteration can be evaluated with the *unordered word error rate* (UWER) as described in Sun and Van hamme (2011a). We compare the performance of the intermediate NMF models with a plain NMF model that is initialized identically but only performs the decomposition described in Section 3.2.1. The bottom panel of Fig. 4 shows 100-*UWER* for the NMF model in NTD.Alt.BW and for the plain NMF model as a function of the total number of NMF iterations. Also here we observe an improvement. Hence we can conclude that alternating optimization of the NTD and HMM benefits both representations.

(3) Parameter sensitivity

In the unsupervised training, no prior information or ground truth labels can be utilized to tune the parameters. Therefore a good model should be robust to the selection of the user-specified parameters. The number of sub-HMMs R and the number of states per sub-HMM L are the two most important parameters in the configuration. We thus report the parameter sensitivity w.r.t. them in Fig. 6. In general, ten states per sub-HMM generate better performance than five states on word discovery on the TIDIGITS database. NTD.Alt.BW is more robust than BW+SA for sensitivity to the number of states. Only for the performance metric of purity and with 10 states, BW+SA approaches the results of NTD.Alt.BW. However, 125 BW iterations were conducted for BW+SA while only 25 iterations for NTD.Alt.BW, making the former a lot slower to train.

The performance of the NTD.Reg.BW with respect to different scales of the regularization parameter λ is shown in Fig. 7. As is expected, only with a large λ NTD.Reg.BW yields significantly different results from BW. The likelihood of NTD.Reg.BW decreases with increasing λ, which is not unexpected since more weight is attributed to NTD which does not maximize the likelihood of the sequential data directly. With large regularization parameters, NTD.Reg.BW

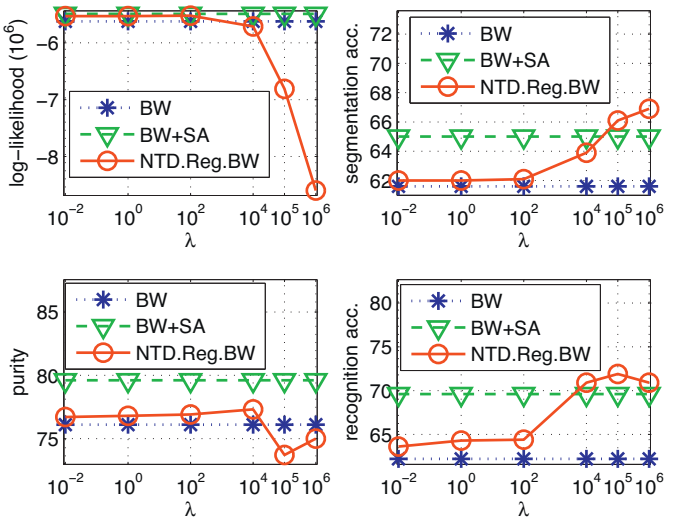

Fig. 7. The performance of NTD regularized BW with respect to the regularization parameter λ.

is observed to outperform BW and BW+SA on segmentation and recognition accuracy. The set of sub-HMMs still contains members that are capable of modeling the digits accurately. Inspection of the mapping matrices also reveals that more garbage sub-HMMs are found for large λ. This is explained by the nature of NTD which does not impose as strictly as BW that the patterns should be contiguous stretches of data. With a large λ NTD.Reg.BW favors NTD. Hence the algorithm tends to generate more sub-HMMs which cannot be linked clearly to a single word and which also explains the lower purity in Fig. 7.

6. Conclusion

A non-negative Tucker decomposition (NTD) model was presented for the unsupervised training of an HMM for sequential pattern discovery, segmentation and recognition. Two training schemes to jointly learn the parameters of the NTD and the HMM were proposed: NTD-regularized Baum–Welch training of the HMM and alternated training of the NTD and the HMM. The second method outperformed the first method, Baum–Welch training and simulated annealing on the task of spoken pattern discovery from real-word speech data. The strength of combined NTD and HMM learning is that NTD explains the data with co-occurrence patterns and therewith provides a global view on the data, while the HMM models the local sequential aspects. A parameter sensitivity study showed some prior knowledge of the number of patterns and their length is required, but that the exact values are not critical.

Acknowledgements

The research was funded by the K.U.Leuven research grant OT/09/028 (VASI). Research of Meng Sun is partially supported by the China Scholarship Council. We would like to thank the editor and the anonymous reviewers for their help to improve the manuscript.

Appendix A. Tensor notations and operations

We consider 3-dimensional tensors in this article, i.e. an array of numbers with three slices. The dimensions of the array are also referred to as *modes*. Mode-1, mode-2 and mode-3 slices and tubes are shown in (a1) of Fig. A.8. The multiplications of a tensor and a matrix are illustrated in (a2). Mode-*i* multiplication entails replacing a mode-*i* tube by the multiplication of the matrix and this mode-*i* tube in the resulting tensor.

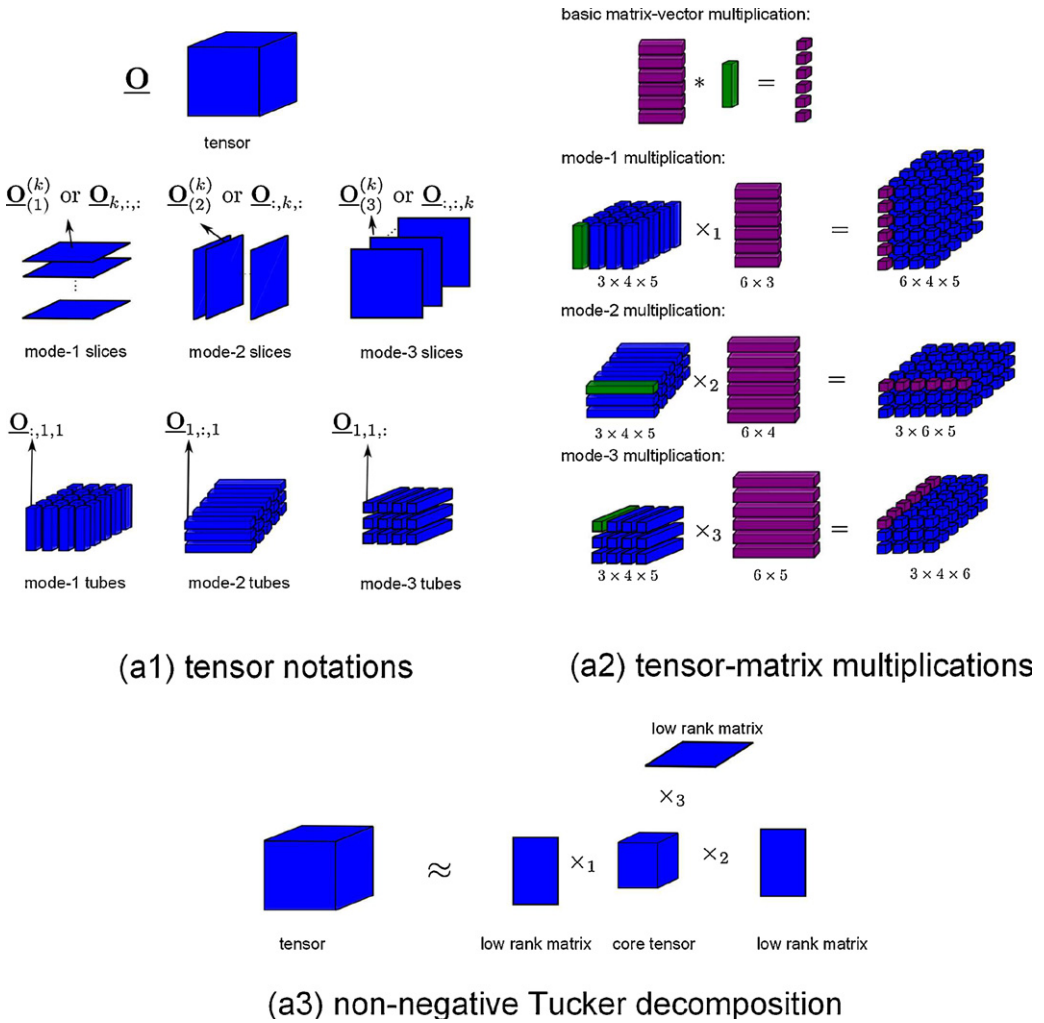

Fig A.8. Tensor notations and operations. (a1) denotes the mode matrices and tubes of a tensor. (a2) shows tensor-matrix multiplications for every mode. (a3) illustrates that the original tensor is decomposed into a low-rank core tensor and three low-rank mode matrices.

Appendix B. Derivation of the NMTF algorithm

To obtain the tri-factorization of the joint probabilistic distribution $\mathbf{Z}$ where $\sum_{i,j} Z_{i,j} = 1$, the chosen cost function is the Kullback–Leibler divergence between $\mathbf{Z}$ and $\mathbf{XYX}'$ with normalization constraints $\sum_i X_{i,k} = 1$ and $\sum_{k,k'} Y_{k,k'} = 1$.

$$\mathcal{F}(\mathbf{X}, \mathbf{Y}) = \sum_{i,j} Z_{i,j} \log \frac{Z_{i,j}}{(XYX')_{i,j}} \tag{B.1}$$

To derive the updating algorithm of $\mathbf{X}$, an auxiliary function is defined firstly with respect to the variable $\mathbf{X}$ with fixed $\mathbf{Y}$. $\mathcal{G}(\mathbf{X}, \mathbf{X}^t)$ is an auxiliary function for $\mathcal{F}(\mathbf{X})$ if

$$\mathcal{G}(\mathbf{X}, \mathbf{X}^t) \geq \mathcal{F}(\mathbf{X}), \quad \mathcal{G}(\mathbf{X}^t, \mathbf{X}^t) = \mathcal{F}(\mathbf{X}^t) \tag{B.2}$$

where $\mathbf{X}^t$ is the solution from the previous iteration. With the convexity of $-\log(.)$, it is easy to see that $\mathcal{G}$ in Eq. (B.3) is an auxiliary function of $\mathcal{F}(\mathbf{X})$ and that $\mathcal{G}(\mathbf{X}, \mathbf{X}^t)$ is a convex function.

$$\mathcal{G}(\mathbf{X}, \mathbf{X}^t) = \sum_{i,j}\left[Z_{i,j}\log Z_{i,j} - \sum_{k,l} Z_{i,j}\frac{X^t_{i,k}Y_{k,l}X^t_{j,l}}{(X^tY(X^t)')_{i,j}}(\log(X_{i,k}Y_{k,l}X_{j,l}) - \log\frac{X^t_{i,k}Y_{k,l}X^t_{j,l}}{(X^tY(X^t)')_{i,j}})\right] \tag{B.3}$$

So the optimization problem with respect to $\mathbf{X}$ is,

$$\begin{aligned} &\min_{\mathbf{X}} \quad \mathcal{G}(\mathbf{X}, \mathbf{X}^t) \\ &s.t. \quad \sum_i X_{i,k} = 1 \end{aligned} \tag{B.4}$$

By using the Lagrange multiplier $\lambda_k(\sum_i X_{i,k} - 1)$, the optimization problem is transformed to solve $\frac{\partial \mathcal{G}(\mathbf{X},\mathbf{X}^t)}{\partial X_{i',k'}} + \lambda_{k'} = 0$ from Eq. (B.5).

$$\begin{aligned} &X^t_{i',k'}\sum_{i,k}\frac{Z_{i,i'}}{(X^tY(X^t)')_{i,i'}}X^t_{i,k}Y_{k,k'} + \frac{Z_{i',i}}{(X^tY(X^t)')_{i',i}}Y_{k',k}X^t_{i,k} + \lambda_{k'}X_{i',k'} = 0 \\ &s.t. \ \sum_{i'} X_{i',k'} = 1 \end{aligned} \tag{B.5}$$

By summing the equations w.r.t. i', one obtains $\lambda_{k'} = -\sum_{i,i',k}(Z_{i,i'}/(X^tY(X^t)')_{i,i'})X^t_{i,k}Y_{k,k'} + (Z_{i',i}/(X^tY(X^t)')_{i',i})Y_{k',k}X^t_{i,k}$. So the final update of $X_{i',k'}$ is

$$X_{i',k'} = X^t_{i',k'}\frac{\sum_{i,k}(Z_{i,i'}/(X^tY(X^t)')_{i,i'})X^t_{i,k}Y_{k,k'} + (Z_{i',i}/(X^tY(X^t)')_{i',i})Y_{k',k}X^t_{i,k}}{\sum_{i,i',k}(Z_{i,i'}/(X^tY(X^t)')_{i,i'})X^t_{i,k}Y_{k,k'}X^t_{i',k'} + (Z_{i',i}/(X^tY(X^t)')_{i',i})Y_{k',k}X^t_{i,k}X^t_{i',k'}} \tag{B.6}$$

The update of $\mathbf{Y}$ is essentially the same as in NMF (Lee and Seung, 2000) by reformulating $\mathbf{Z} \approx \mathbf{XYX}'$ to be $\mathrm{vec}(\mathbf{Z}') \approx (\mathbf{X} \otimes \mathbf{X})\mathrm{vec}(\mathbf{Y}')$. *vec* is to transform a matrix to a vector by concatenating its columns, and $\otimes$ is the Kronecker product. Following the property from NMF, given $\sum_\mu (X \otimes X)_{\mu,\gamma}$=1, $\sum_\mu(\mathrm{vec}(Z'))_\mu = \sum_\kappa(\mathrm{vec}(Y'))_\kappa$ holds. $\sum_\mu (X \otimes X)_{\mu,\gamma}$=1 is guaranteed by $\sum_i X_{i,k} = 1$, thus $\sum_{k,l} Y_{k,l} = \sum_{i,j} Z_{i,j} = 1$. The update of $\mathbf{Y}$ in the form of tri-factorization is given in Eq. (B.7) where $\mathbf{Y}^t$ is from the previous iteration.

$$Y_{k,l} = Y^t_{k,l}\sum_{i,i'}\frac{Z_{i,i'}}{(X^tY(X^t)')_{i,i'}}X_{i,k}X_{i',l} \tag{B.7}$$

References

Baum, L.E., Petrie, T., Soules, G., Weiss, N., 1970. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. The Annals of Mathematical Statistics 41 (1), 164–171.

Berry, M.W., Browne, M., Langville, A.N., Pauca, V.P., Plemmons, R.J., 2006. Algorithms and applications for approximate nonnegative matrix factorization. Computational Statistics and Data Analysis, 155–173.

Blei, D.M., Moreno, P.J., 2001. Topic segmentation with an aspect hidden Markov model. In: SIGIR, pp. 343–348.

Boves, L., ten Bosch, L., Moore, R., 2007. ACORNS—towards computational modeling of communication and recognition skills. In: International Conference on Cognitive Informatics, pp. 349–356.

Cichocki, A., Zdunek, R., Phan, A.H., Amari, S., 2009. Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation. Wiley.

Clemente, I.A., Heckmann, M., Denecke, A., Wrede, B., Goerick, C., 2010. Incremental word learning using large-margin discriminative training and variance floor estimation. In: INTERSPEECH, pp. 889–892.

Cybenko, G., Crespi, V., 2011. Learning hidden Markov models using non-negative matrix factorization. IEEE Transactions on Information Theory 57 (6), 3963–3970.

Donoho, D., Stodden, V., 2004. When does non-negative matrix factorization give a correct decomposition into parts? In: NIPS. MIT Press.

Driesen, J., Gemmeke, J.F., Van hamme, H., 2012. Weakly supervised keyword learning using sparse representations of speech. In: ICASSP, pp. 5145–5148.

Fevotte, C., Cemgil, A.T., 2009. Nonnegative matrix factorizations as probabilistic inference in composite models. In: EUSIPCO.

Finesso, L., Grassi, A., Spreij, P., 2010. Approximation of stationary processes by hidden Markov models. Mathematics of Control, Signals, and Systems 22, 1–22.

Gaussier, E., Goutte, C., 2005. Relation between PLSA and NMF and implications. In: SIGIR, pp. 601–602.
Heckmann, M., Brandl, H., Domont, X., Vaz, M., Schmuedderich, J., Bolder, B., Joublin, F., Goerick, C., 2009. Language acquisition embedded into Tutor–Robot Interaction. In: Proc. ACORNS Workshop Computational Models of Language Evolution, Acquisition and Processing, ACORNS, Brighton.
Hirsch, H., Pearce, D., 2000. The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In: Proceedings of ISCA ITRW ASR, pp. 181–188.
Hoffman, T., 1999. Probabilistic latent semantic analysis. In: Uncertainty in Artificial Intelligence.
Hsu, D., Kakade, S.M., Zhang, T., 2011. A spectral algorithm for learning hidden Markov models. Journal of Computer and System Sciences 78 (5), 1460–1480.
Hu, J., Brown, M.K., Turin, W., 1996. HMM based on-line handwriting recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 18, 1039–1045.
Huang, X., Acero, A., Hon, H.-W., 2001. Spoken Language Processing: A Guide to Theory Algorithm, and System Development, 1st ed. Prentice Hall PTR, Upper Saddle River, NJ, USA, ISBN 0130226165.
Jansen, A., Church, K., 2011. Towards unsupervised training of speaker independent acoustic models. In: INTERSPEECH, pp. 1692–1693.
Jansen, A., Church, K., Hermansky, H., 2010. Towards spoken term discovery at scale with zero resources. In: INTERSPEECH, pp. 1676–1679.
Johnson, M., 2007. Why doesn't EM find good HMM POS-taggers? In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pp. 296–305.
Juang, B.H., Rabiner, L.R., 1990. The segmental K-means algorithm for estimating parameters of hidden Markov Models. IEEE Transactions on Acoustic, Speech, and Signal Processing 38, 1639–1641.
Khreich, W., Granger, E., Miri, A., Sabourin, R., 2012. A survey of techniques for incremental learning of HMM parameters. Information Science 197, 105–130.
Lakshminarayanan, B., Raich, R., 2010. Non-negative matrix factorization for parameter estimation in hidden Markov models. In: IEEE International Workshop on Machine Learning for Signal Processing (MLSP), pp. 89–94.
Lee, D.D., Seung, H.S., 1999. Learning the parts of objects by non-negative matrix factorization. Nature 401, 788–791.
Lee, D.D., Seung, H.S., 2000. Algorithms for non-negative matrix factorization. In: NIPS, pp. 556–562.
Levinson, S., Rabiner, L., Sondhi, M., 1982. An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. The Bell System Technical Journal 62, 1035–1074.
Mohri, M., Pereira, F., Riley, M., 2002. Weighted finite-state transducers in speech recognition. Computer Speech and Language 16 (1), 69–88.
Mørup, M., Hansen, L.K., Arnfred, S.M., 2008. Algorithms for sparse non-negative tucker. Neural Computation 20, 2112–2131.
Park, A.S., Glass, J.R., 2008. Unsupervised pattern discovery in speech. IEEE Transactions on Audio, Speech and Language Processing 16 (1), 186–197.
Paul, D.B., 1985. Training of HMM recognizers by simulated annealing. In: ICASSP, pp. 13–16.
Pedersen, J.S., Hein, J., 2003. Gene finding with a hidden Markov model of genome structure and evolution. Bioinformatics 19, 219–227.
Rabiner, L.R., 1989. A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE 77 (2), 257–286.
Roweis, S., 1999. http://cs.nyu.edu/roweis/code.html
Roy, D., 2003. Grounded spoken language acquisition: experiments in word learning. IEEE Transactions on Multimedia 5 (2), 197–209.
Siu, M.-H., Gish, H., Lowe, S., Chan, A., 2011. Unsupervised audio patterns discovery using HMM-based self-organized units. In: INTERSPEECH, pp. 2333–2336.
Smith, N.A., 2006. Novel Estimation Methods for Unsupervised Discovery of Latent Structure in Natural Language Text. Ph.D. Thesis, Johns Hopkins University, 2006.
Song, L., Siddiqi, S.M., Gordon, G.J., Smola, A.J., 2010. Hilbert space embeddings of hidden Markov Models. In: ICML, pp. 991–998.
Sun, M., Van hamme, H., 2011a. Unsupervised vocabulary discovery using non-negative matrix factorization with graph regularization. In: ICASSP, pp. 5152–5155.
Sun, M., Van hamme, H., 2011b. A two-layer non-negative matrix factorization model for vocabulary discovery. In: Symposium on Machine Learning in Speech and Language Processing.
Sundaram, S., Bellegarda, J.R., 2010. Latent perceptual mapping: a new acoustic modeling framework for speech recognition. In: INTERSPEECH, pp. 881–884.
ten Bosch, L., Van hamme, H., Boves, L., 2008. A computational model of language acquisition: focus on word discovery. In: INTERSPEECH, pp. 2570–2573.
Tucker, L.R., 1966. Some mathematical notes on three-mode factor analysis. Psychometrika, 279–311.
Van hamme, H., 2008. HAC-models: a novel approach to continuous speech recognition. In: INTERSPEECH, pp. 2554–2557.
Vanluyten, B., Willems, J., De Moor, B., 2008. Structured nonnegative matrix factorization with applications to hidden Markov realization and clustering. Linear Algebra and its Applications 429, 1409–1424.
Varadarajan, B., Khudanpur, S., Dupoux, E., 2008. Unsupervised learning of acoustic sub-word units. In: ACL, pp. 165–168.
Wu, C.F.J., 1983. On the convergence properties of the EM algorithm. The Annals of Statistics 11 (1), 95–103.
Yang, Z., Oja, E., 2011. Quadratic non-negative matrix factorization. Pattern Recognition, 1–11.
Zhang, Y., Glass, J., 2010. Towards multi-speaker unsupervised speech pattern discovery. In: in: ICASSP, pp. 4366–4369.
Zhuang, X., Huang, J.T., Hasegawa-Johnson, M., 2009. Speech retrieval in unknown languages: a pilot study. In: Proceedings of the Third International Workshop on Cross Lingual Information Access: Addressing the Information Need of Multilingual Societies, CLIAWS3 '09, pp. 3–11.