Active Learning of Partially Hidden Markov Models

Tobias Scheffer and Stefan Wrobel
University of Magdeburg, FIN/IWS, PO Box 4120, 39016 Magdeburg, Germany
{scheffer, hoche, wrobel}@iws.cs.uni-magdeburg.de

July 24, 2001

Abstract

We consider the task of learning hidden Markov models (HMMs) when only *partially (sparsely) labeled* observation sequences are available for training. This setting is motivated by the information extraction problem, where only few tokens in the training documents are given a semantic tag while most tokens are unlabeled. We first describe the partially hidden Markov model together with an algorithm for learning HMMs from partially labeled data. We then present an active learning algorithm that selects "difficult" unlabeled tokens and asks the user to label them. We study empirically by how much active learning reduces the required data labeling effort, or increases the quality of the learned model achievable with a given amount of user effort.

1 Introduction

Given the enormous amounts of information available only in unstructured or semi-structured textual documents, tools for *information extraction* (IE) have become enormously important (see [6, 5] for an overview). IE tools identify the relevant information in such documents and convert it into a structured format such as a database or an XML document [2]. While first IE algorithms were hand-crafted sets of rules (*e.g.,* [9]), researchers soon turned to learning extraction rules from hand-labeled documents (*e.g.,* [10, 12, 8]). Unfortunately, rule-based approaches sometimes fail to provide the necessary robustness against the inherent variability of document structure, which has led to the recent interest in the use of hidden Markov models (HMMs) [17, 14] for this purpose.

HMMs are stochastic automata that move within a finite set of states, emitting an observation in each step. Each state has a distinct distribution over the possible observations but, in general, it is not possible to uniquely identify the state that a given observation was generated in. The well-known Viterbi algorithm finds the sequence of states that is most likely to have generated a given observation sequence. The Baum-Welch algorithm, an instantiation of EM, can be used to estimate the most likely HMM parameters given a collection of observation sequences. Speech recognition [11] and computational biochemistry [1] are well-known applications of HMMs.

(Non-hidden) Markov model algorithms that are used for part-of-speech tagging [3] and for information extraction [14] require each observation (*i.e.,* token) of the observation sequences (documents) used for training to be labeled with the state (corresponding to a part of speech) in which it was generated. In this case, as the state is not hidden, finding the MM parameters is a simple probability observation problem.

In the information extraction problem, some tokens of the example observation sequences (those which are to be extracted) are labeled by the user, the remaining tokens are not. In order to be able to exploit the labeling information and yet be able to handle the unlabeled tokens, we propose the *partially hidden Markov model* (PHMM) for this task.

The paper is organized as follows. We introduce our terminology in Section 2, and present the partially hidden Markov model with our modified forward-backward and Baum-Welsh algorithms in Section 3. In Section 4, we discuss our active learning procedure; we present our results in Section 5. Section 6 concludes.

2 Preliminaries

Hidden Markov models (see, [15] for an introduction) are a very robust statistical method for analysis of temporal data. An HMM $\lambda = (\pi, a, b)$ consists of finitely many states $\{S_1, \ldots, S_N\}$ with probabilities $\pi_i = P(q_1 = S_i)$, the probability of starting in state S_i, and $a_{ij} = P(q_{t+1} = S_j | q_t = S_i)$, the probability of a transition from state S_i to S_j. Each state is characterized by a probability distribution $b_i(O_t) = P(O_t | q_t = S_i)$ over observations. In the information extraction context, an observation is typically a token. The labels X_i correspond to the n target states $S_1, \ldots, S_n$ of the HMM. Background tokens without label are emitted in all HMM states $S_{n+1}, \ldots, S_N$ which are not one of the target states.

We might, for instance, want to convert an HTML phone directory into a database using an HMM with four nodes (labeled *name, firstname, phone, none*). The observation sequence "John Smith, extension 7343" would then correspond to the state sequence (*firstname, name, none, phone*).

HMM algorithms are described in terms of some recursively computable probabilities. We briefly describe some elements of these algorithms (those that we need for our learning algorithm in the next section) and refer to [15] for a more detailed discussion. $\alpha_t(i) = P(q_t = S_i, O_1, \ldots, O_t | \lambda)$ is called the forward variable and quantifies the probability of reaching state S_i at time t and observing the initial part $O_1, \ldots, O_t$ of the observation sequence. Of course, in this definition it is assumed that no information except for the observation sequence is given about the state.

$\beta_t(i) = P(O_{t+1} \ldots O_T | q_t = S_i, \lambda)$ is called the backward variable and quantifies the chance of observing the rest sequence $O_{t+1}, \ldots, O_T$ when we are in state S_i at time t. $\alpha_t(i)$ and $\beta_t(i)$ can be computed recursively by the *forward-backward* procedure (see [15]).

$\gamma_t(i) = P(q_t = S_i | O, \lambda)$ is the probability of being in state S_i at time t given observation sequence $O = (O_1, \ldots, O_T)$; γ can easily be calculated from α and β. Using the forward backward algorithm we can determine the state that are most likely (maximize $\gamma_t(i)$) given an observation sequence O, and thus "apply" an HMM with known parameters λ to a new observation sequence O.

The Baum-Welsh algorithm can be used to estimate the most likely model parameters given a set of observation sequences $\mathcal{O} = \{O_1(s), \ldots, O_{T_s}^{(s)}\}$. In the standard HMM setting (*e.g.,* in speech recognition), only observation sequences but no corresponding states are given. In the standard Markov model setting (*e.g.,* in part-of-speech tagging), the state sequences that corresponds to these observation sequences are known. By contrast, in the setting that we study in this paper, a labeling function is known, that defines, for each observation, a set of possible states. For any observation, this set can contain one (part-of-speech tagging), some, or all states (speech recognition).

In the information extraction problem, some tokens are typically attached a semantic tag; the state of these tokens is uniquely determined. Other tokens are marked as "background", and it is known that these tokens have not been generated in one of the "target" states, but any background state is possible. Token can also be left unmarked which says nothing about their state.

Note that the problem of learning from partially labeled observation sequences is substantially different from both learning from unlabeled and learning from completely labeled documents and no known HMM learning algorithm can be applied. The Baum-Welch algorithm that learns from unlabeled documents does not map the observations that are to be extracted to any particular target state which is crucial for information extraction. Simple probability estimation used for (non-hidden) Markov models cannot be applied because the state corresponding to the unlabeled observations is not known.

3 Partially Hidden Markov Models

What is needed to adapt the Baum-Welch procedure to partially labeled data? By looking at the parameter re-estimation formula used in the algorithm, we see that it centrally depends on being able to estimate the probability of being in state S_i at time t and in state S_j at time $t+1$, given the observations and the model:

$$\xi_t(i,j) = P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda) \tag{1}$$

Let us first define the set σ_t of states in which observation O_t could have been generated.

$$\Sigma = (\sigma_1, \ldots, \sigma_T); \quad \sigma_i \subseteq \{1, \ldots, N\} \tag{2}$$

$$\tag{3}$$

σ_t is given by the labels of a training document. When an observation O_t is labeled with a unique state in which it must have been generated, then σ_t contains that single state. When O_t has no label, then σ_t contains all possible states.

The forward-backward algorithm is used to determine the values of the variables $\alpha_t(i)$, $\beta_t(i)$, and $\gamma_t(i)$. We will first show we can modify the definitions of these variables and the algorithm to take partial labeling information into account. Our considerations lead to the new "backward-forward-backward" algorithm.

We first have to introduce a new set of probabilities τ not found in the standard setting for unlabeled data. $\tau_t(i)$ is the probability of observing the remaining labeled states, beginning at step t, given that we are in state S_i at time t and given the model (Equation 4).

$$\tau_t(i) = P(\sigma_{t+1}, \ldots, \sigma_T \mid q_t = S_i, \lambda) \tag{4}$$

The following lemma shows how to compute τ.

Lemma 1 (Computation of τ) *Given an HMM with parameters λ, observations $O = O_1, \ldots, O_T$, and corresponding possible states $\Sigma = (\sigma_1, \ldots, \sigma_T)$, for any $1 \leq t \leq T$, and $1 \leq i \leq N$, we can compute $\tau_t(i) = P(\sigma_{t+1}, \ldots, \sigma_T \mid q_t = S_i, \lambda)$ as given by Equations 5 and 6.*

$$\tau_T(i) = 1 \tag{5}$$

$$\tau_t(i) = \sum_{j \in \sigma_{t+1}} \tau_{t+1}(j) a_{ij} \tag{6}$$

Proof. (a) Base case. Note that $t = T - 1$ is actually the base case as $\tau_T(i)$ is not a proper probability.

$$\begin{aligned} P(\sigma_T | q_{T-1} = S_i, \lambda) &= P(\sigma_T | q_{T-1} = S_i, \lambda) && (7) \\ &= \sum_{j \in \sigma_T} P(q_T = S_j | q_{T-1} = S_i, \lambda) && (8) \\ &= \sum_{j \in \sigma_T} a_{ij} = \sum_{j \in \sigma_T} \tau_T(j) a_{ij} && (9) \end{aligned}$$

q_T lies in σ_T when it is one of the states $j \in \sigma_T$ (Equation 8). $(q_T = S_j | q_{T-1} = S_i, \lambda)$ is just the transition probability a_{ij}; since $\tau_T(i)$ is defined to be 1 (Equation 5), we can rewrite the term in the recursive form of Equation 6.

(b) Inductive case. In Equation 11 we sum over all next states q_{t+1}. Note that $P(q_{t+1} = S_j, \sigma_{t+1}, \ldots)$ must be zero if $j \notin \sigma_{t+1}$, so we drop those terms in Equation 12. Now, $P(q_{t+1} = S_j, \sigma_{t+1}, \ldots) = P(q_{t+1} = S_j, \ldots)$ because S_j is known to lie in σ_{t+1} (Equation 12). We factorize the transition probability in Equation 13; the residual has the required recursive structure (Equation 14).

$$\begin{aligned} \tau_t(i) &= P(\sigma_{t+1}, \ldots, \sigma_T \mid q_t = S_i, \lambda) && (10) \\ &= \sum_{j=1}^{N} P(q_{t+1} = S_j, \sigma_{t+1}, \ldots, \sigma_T \mid q_t = S_i, \lambda) && (11) \\ &= \sum_{j \in \sigma_{t+1}} P(q_{t+1} = S_j, \sigma_{t+2}, \ldots, \sigma_T \mid q_t = S_i, \lambda) && (12) \\ &= \sum_{j \in \sigma_{t+1}} P(\sigma_{t+1}, \ldots, \sigma_T \mid q_{t+1} = S_j, q_t = S_i, \lambda) P(q_{t+1} = S_j | q_t = S_i, \lambda) && (13) \\ &= \sum_{j \in \sigma_{t+1}} \tau_{t+1}(j) a_{ij} && (14) \end{aligned}$$

□

Note that we can express $P(\Sigma|\lambda)$ in terms of τ (Equation 16).

$$\begin{aligned} P(\Sigma|\lambda) &= P(\sigma_1, \ldots, \sigma_T | \lambda) && (15) \\ &= \sum_{i \in \sigma_1} P(\sigma_1, \ldots, \sigma_T | q_1 = S_i \lambda) \pi_i = \sum_{i \in \sigma_1} \tau_1(i) \pi_i && (16) \end{aligned}$$

We can now define and calculate the $\alpha_t(i)$ variable.

Lemma 2 (Computation of α) *Given a HMM with parameters λ, and observations $O = O_1, \ldots, O_T$, with corresponding possible states Σ, for any $1 \le t \le T$, and $1 \le j \le N$, we can compute $\alpha_t(j) = P(q_t = S_j, O_1, \ldots, O_t \mid \Sigma, \lambda)$ as in Equation 17 and 18.*

$$\alpha_1(j) = b_j(O_1) \pi_j \frac{\tau_1(j)}{\sum_{i \in \sigma_1} \tau_1(i) \pi_i} \qquad (17)$$

$$\alpha_{t+1}(j) = \begin{cases} \sum_{i \in \sigma_t} \left(\alpha_t(i) a_{ij} \frac{\tau_{t+1}(j)}{\tau_t(i)} \right) b_j(O_{t+1}) & \text{if } j \in \sigma_{t+1} \\ 0 & \text{otherwise} \end{cases} \qquad (18)$$

Proof. (a) Initialization: $t = 1$. We factorize $P(q_1 = S_i|\Sigma, \lambda)$ in Equation 20. Under the Markov assumption for observations, the residual is just the output probability $b_i(O_1)$ (Equation 21). In Equation 21 we also apply Bayes' equation. In Equation 22 we apply Equation 16.

$$\begin{aligned}
\alpha_1(i) &= P(O_1, q_1 = S_i|\Sigma, \lambda) &(19)\\
&= P(O_1|q_1 = S_i, \Sigma, \lambda) P(q_1 = S_i|\Sigma, \lambda) &(20)\\
&= b_i(O_1) P(\Sigma|q_1 = S_i, \lambda) \frac{\pi_1(i)}{P(\Sigma|\lambda)} &(21)\\
&= b_i(O_1)\tau_1(i) \frac{\pi_1(i)}{\sum_{j\in\sigma_1}^{N} \tau_1(j)\pi_j} &(22)
\end{aligned}$$

(b) Recursive case. In Equation 24 we sum over all previous states q_t. $P(q_t = S_i, q_{t+1} = S_j, \ldots|\Sigma, \lambda)$ is zero for $q_t \notin \sigma_t$, so we drop these terms from the sum in Equation 25. In Equation 26 we split up the probability into three components: $P(q_t = S_i, O_1, \ldots, O_t|\Sigma, \lambda)$ is just $\alpha_t(i)$ (Equation 27); $P(O_{t+1}|q_{t+1} = S_j, q_t = S_i, O_1, \ldots, O_t, \Sigma, \lambda)$ is, under the Markov assumption for observations, the output probability $b_j(O_{t+1})$. The residual ($P(q_{t+1} = S_j|q_t = S_i, O_1, \ldots, O_t, \Sigma, \lambda)$) is broken up in Equations 29 through 31.

$$\begin{aligned}
\alpha_{t+1}(j) &= P(q_{t+1} = S_j, O_1, \ldots, O_{t+1}|\Sigma, \lambda) &(23)\\
&= \sum_{i=1}^{N} P(q_t = S_i, q_{t+1} = S_j, O_1, \ldots, O_{t+1}|\Sigma, \lambda) &(24)\\
&= \sum_{i\in\sigma_t} P(q_t = S_i, q_{t+1} = S_j, O_1, \ldots, O_{t+1}|\Sigma, \lambda) &(25)\\
&= \sum_{i\in\sigma_t} P(q_t = S_i, O_1, \ldots, O_t|\Sigma, \lambda) P(q_{t+1} = S_j|q_t = S_i, O_1, \ldots, O_t, \Sigma, \lambda)\\
&\quad P(O_{t+1}|q_{t+1} = S_j, q_t = S_i, O_1, \ldots, O_t, \Sigma, \lambda) &(26)\\
&= \sum_{i\in\sigma_t} \alpha_t(i) P(q_{t+1} = S_j|q_t = S_i, O_1, \ldots, O_t, \Sigma, \lambda) b_j(O_{t+1}) &(27)\\
&= \begin{cases} \sum_{i\in\sigma_t} \left(\alpha_t(i) a_{ij} \frac{\tau_{t+1}(j)}{\tau_t(i)}\right) b_j(O_{t+1}) & \text{if } j \in \sigma_{t+1}\\ 0 & \text{otherwise}\end{cases} &(28)
\end{aligned}$$

In Equation 29, we apply Bayes' equation, moving $\sigma_{t+1}, \ldots, \sigma_T$ to the front and $q_{t+1} = S_j$ into the conditional part. Under the Markov assumption for transitions, $P(q_{t+1} = S_j|q_t = S_i, O_1, \ldots, O_t, \sigma_1, \ldots, \sigma_t, \lambda)$ is just a_{ij} (Equation 31). $P(\sigma_{t+1}, \ldots, \sigma_T|q_t = S_i, O_1, \ldots, O_t, \sigma_1, \ldots, \sigma_t, \lambda)$ equals $\tau_t(i)$ according to the definition of τ. Let us now look at $P(\sigma_{t+1}, \ldots, \sigma_T|q_{t+1} = S_j, q_t = S_i, O_1, \ldots, O_t, \sigma_1, \ldots, \sigma_t, \lambda)$. We have $q_{t+1} = S_j$ in the conditional part and σ_{t+1} as random variable. If $q_{t+1} = S_j \notin \sigma_{t+1}$, the this probability must be zero; if we have $q_{t+1} = S_j$ with $j \in \sigma_{t+1}$, then we can drop σ_{t+1} from the left hand side of this probability because it is true with certainty. We then obtain $P(\sigma_{t+2}, \ldots, \sigma_T|q_{t+1} = S_j, q_t = S_i, O_1, \ldots, O_t, \sigma_1, \ldots, \sigma_t, \lambda)$ which equals $\tau_{t+1}(j)$. Hence, we arrive at Equation 31. If we apply this result to Equation 27, we obtain Equation 28 which completes the proof.

$$\begin{aligned}
&P(q_{t+1} = S_j|q_t = S_i, O_1, \ldots, O_t, \Sigma, \lambda) &(29)\\
&\quad = P(\sigma_{t+1}, \ldots, \sigma_T|q_{t+1} = S_j, q_t = S_i, O_1, \ldots, O_t, \sigma_1, \ldots, \sigma_t, \lambda)
\end{aligned}$$

$$
\frac{P(q_{t+1} = S_j | q_t = S_i, O_1, \ldots, O_t, \sigma_1, \ldots, \sigma_t, \lambda)}{P(\sigma_{t+1}, \ldots, \sigma_T | q_t = S_i, O_1, \ldots, O_t, \sigma_1, \ldots, \sigma_t, \lambda)} \tag{30}
$$

$$
= \begin{cases} \frac{\tau_{t+1}(j)}{\tau_t(i)} a_{ij} & \text{if } j \in \sigma_{t+1} \\ 0 & \text{otherwise} \end{cases} \tag{31}
$$

□

Lemma 3 (Computation of β) *Given a HMM with parameters λ, and observations $O = O_1, \ldots, O_T$, with corresponding possible states Σ, for any $1 \leq t \leq T$, and $1 \leq i \leq N$, we can compute $\beta_t(i) = P(O_{t+1}, \ldots, O_T \mid q_t = S_i, \Sigma, \lambda)$ as in Equations 32 and 33.*

$$
\beta_T(i) = 1 \tag{32}
$$

$$
\beta_t(i) = \sum_{j \in \sigma_{t+1}} \left(\beta_{t+1}(j) a_{ij} \frac{\tau_{t+1}(j)}{\tau_t(i)} \right) b_j(O_{t+1}) \tag{33}
$$

Note that $\beta_t(i)$ is undefined when $i \notin \sigma_t$ (because then $\tau_t(i) = 0$). Intuitively, the corresponding probability of observations $O_{t+1}, \ldots, O_T$ when in an impossible state $q_t \notin \sigma_t$ should not be defined.

Proof. (a) Again, $t = T - 1$ is the base case as $\beta_T(i)$ does not possess a random variable and is not a probability. In Equation 35 we sum over all possible final states S_j. We marginalize $P(O_T | q_T = S_j, q_{T-1} = S_i, \Sigma, \lambda)$ (which, under the Markov assumption for observations is $b_j(O_T)$) in Equation 36. The residual $P(q_T = S_j | q_{T-1} = S_i, \sigma_1, \ldots, \sigma_T, \lambda)$ is broken up in Equations 39 and 41.

$$
\begin{aligned}
\beta_{T-1}(i) &= P(O_T, | q_{T-1} = S_i, \Sigma, \lambda) && (34) \\
&= \sum_{j \in \sigma_T} P(O_T, q_T = S_j | q_{T-1} = S_i, \Sigma, \lambda) && (35) \\
&= \sum_{j \in \sigma_T} P(O_T | q_T = S_j, q_{T-1} = S_i, \Sigma, \lambda) \cdot P(q_T = S_j | q_{T-1} = S_i, \Sigma, \lambda) && (36) \\
&= \sum_{j \in \sigma_T} b_j(O_T) \cdot P(q_T = S_j | q_{T-1} = S_i, \sigma_1, \ldots, \sigma_T, \lambda) && (37) \\
&= \sum_{j \in \sigma_T} \left(\beta_T(j) \frac{\tau_T(j)}{\tau_{T-1}(i)} \right) b_j(O_T) && (38)
\end{aligned}
$$

In Equation 39 we apply Bayes' rule. We known that $P(q_T = S_j | q_{T-1} = S_i, \sigma_1, \ldots, \sigma_T, \lambda) = 1$ because we are summing only S_j with $j \in \sigma_T$. Under the Markov assumption for transitions, $P(q_T = S_j | q_{T-1} = S_i, \sigma_1, \ldots, \sigma_{T-1}, \lambda) = a_{ij}$. Applying this to Equation 36 and introducing $\beta_T(j)$ as additional factor (defined to be 1) leads to the desired result in Equation 38.

$$
\begin{aligned}
&P(q_T = S_j | q_{T-1} = S_i, \sigma_1, \ldots, \sigma_T, \lambda) \\
&\quad = P(\sigma_T | q_T = S_j, q_{T-1} = S_i, \sigma_1, \ldots, \sigma_{T-1}, \lambda) \\
&\qquad \frac{P(q_T = S_j | q_{T-1} = S_i, \sigma_1, \ldots, \sigma_{T-1}, \lambda)}{P(\sigma_T | q_{T-1} = S_i, \sigma_1, \ldots, \sigma_{T-1}, \lambda)} && (39) \\
&\quad = P(\sigma_T | q_T = S_j, q_{T-1} = S_i, \sigma_1, \ldots, \sigma_{T-1}, \lambda) \frac{a_{ij}}{\tau_{T-1}(i)} && (40) \\
&\quad = a_{ij} \frac{\tau_T(j)}{\tau_{T-1}(i)} && (41)
\end{aligned}
$$

(b) Recursive case. We sum over all possible successor states S_j in Equation 43 (the summands for $j \notin \sigma_{t+1}$ are zero). Now we marginalize $P(O_{t+2}, \ldots, O_T | O_{t+1}, q_{t+1} = S_j, q_t = S_i, \Sigma, \lambda)$ (which is $\beta_{t+1}(j)$ under the Markov assumption) and $P(O_{t+1} | q_{t+1} = S_j, q_t = S_i, \Sigma, \lambda)$ (which is $b_j(O_{t+1})$ under Markov assumption). The residual of Equation 45 is broken up in Equations 47 through 49.

$$
\begin{aligned}
\beta_t(i) &= P(O_{t+1}, \ldots, O_T | q_t = S_i, \Sigma, \lambda) && (42)\\
&= \sum_{j \in \sigma_{t+1}} P(O_{t+1}, \ldots, O_T, q_{t+1} = S_j | q_t = S_i, \Sigma, \lambda) && (43)\\
&= \sum_{j \in \sigma_{t+1}} P(O_{t+2}, \ldots, O_T | O_{t+1}, q_{t+1} = S_j, q_t = S_i, \Sigma, \lambda) \cdot \\
&\quad P(O_{t+1} | q_{t+1} = S_j, q_t = S_i, \Sigma, \lambda) \cdot P(q_{t+1} = S_j | q_t = S_i, \Sigma, \lambda) && (44)\\
&= \sum_{j \in \sigma_{t+1}} \beta_{t+1}(j) \cdot b_j(O_{t+1}) \cdot P(q_{t+1} = S_j | q_t = S_i, \sigma_1, \ldots, \sigma_T, \lambda) && (45)\\
&= \sum_{j \in \sigma_{t+1}} \left(\beta_{t+1}(j) \frac{\tau_{t+1}(j)}{\tau_t(i)} \right) b_j(O_{t+1}) && (46)
\end{aligned}
$$

Again, we apply Bayes' equation in Equation 47. Since j is known to be in σ, $P(\sigma_{t+1}, \ldots, \sigma_T | \ldots)$ equals $P(\sigma_{t+2}, \ldots, \sigma_T | \ldots)$ (Equation 48). $P(q_{t+1} = S_j | q_t = S_i, \sigma_1, \ldots, \sigma_t, \lambda)$ equals a_{ij} under the Markov assumption; the denominator equals $\tau_t(i)$. The resulting Equation 49 takes us from Equation 45 to Equation 46.

$$
\begin{aligned}
&P(q_{t+1} = S_j | q_t = S_i, \sigma_1, \ldots, \sigma_T, \lambda) \\
&\quad = P(\sigma_{t+1}, \ldots, \sigma_T | q_{t+1} = S_j, q_t = S_i, \sigma_1, \ldots, \sigma_t, \lambda) \\
&\qquad \frac{P(q_{t+1} = S_j | q_t = S_i, \sigma_1, \ldots, \sigma_t, \lambda)}{P(\sigma_{t+1}, \ldots, \sigma_T | q_t = S_i, \sigma_1, \ldots, \sigma_t, \lambda)} && (47)\\
&\quad = P(\sigma_{t+2}, \ldots, \sigma_T | q_{t+1} = S_j, q_t = S_i, \sigma_1, \ldots, \sigma_t, \lambda) \frac{a_{ij}}{\tau_t(i)} && (48)\\
&\quad = a_{ij} \frac{\tau_{t+1}(j)}{\tau_t(i)} && (49)
\end{aligned}
$$

□

Lemma 4 (Computation of γ) *Given a HMM with parameters λ and observations $O = O_1, \ldots, O_T$ with corresponding possible states Σ, for any $1 \leq t \leq T$, and $1 \leq i \leq N$, we can compute $\gamma_t(i) = P(q_t = S_i \mid O, \Sigma, \lambda)$ as in Equation 50.*

$$
\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\Sigma, \lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j \in \sigma_T} \alpha_T(i)} \tag{50}
$$

Proof. In Equation 52 we convert the conditional into the joint probability. We marginalize $P(q_t = S_i, O_1, \ldots, O_t | \Sigma, \lambda)$ in Equation 53. The Markov assumption takes us to Equation 54. The definitions of α and β result in Equation 55.

$$
\begin{aligned}
\gamma_t(i) &= P(q_t = S_i | O, \Sigma, \lambda) && (51)\\
&= \frac{P(q_t = S_i, O_1, \ldots, O_T | \Sigma, \lambda)}{P(O|\Sigma, \lambda)} && (52)
\end{aligned}
$$

$$= \frac{P(q_t = S_i, O_1, \ldots, O_t|\Sigma, \lambda)P(O_{t+1}, \ldots, O_T|q_t = S_i, O_1, \ldots, O_t, \Sigma, \lambda)}{P(O|\Sigma, \lambda)} \quad (53)$$

$$= \frac{P(q_t = S_i, O_1, \ldots, O_t|\Sigma, \lambda)P(O_{t+1}, \ldots, O_T|q_t = S_i, \Sigma, \lambda)}{P(O|\Sigma, \lambda)} \quad (54)$$

$$= \frac{\alpha_t(i)\beta_t(i)}{P(O|\Sigma, \lambda)} \quad (55)$$

$$(56)$$

□

We are now ready to present the new backward-forward-backward algorithm (Table 1) for computing the α, β, and γ variables, given an HMM λ, an observation sequence $O = O_1, \ldots, O_T$, and a sequence of possible states $\Sigma = \sigma_1, \ldots, \sigma_T$. Table 2 describes the known forward-backward algorithm, a special case of backward-forward-backward where no labels (*i.e.,* constraints on the possible states for each observations) are available. This is the typical case when a model λ is applied to extract information from a new, unlabeled document, while backward-forward-backward will be called by the EM algorithm during training, when partially labeled documents are available.

Table 1: Backward-Forward-Backward algorithm

1. **Input** Observation sequence $(O_1, \ldots, O_T)$, possible corresponding states $\sigma_1, \ldots, \sigma_T$, HMM λ.
2. **For all** i, **Let** $\tau_T(i) = 1$ according to Equation 5.
3. (first backward step) **For** $t = T_1, \ldots, 1$, **For all** $i = 1, \ldots, N$ **Determine** $\tau_t(i)$ according to Equation 6.
4. **For all** i, **Determine** $\alpha_1(i)$ according to Equation 17; **Determine** $\beta_T(i)$ according to Equation 32.
5. (forward step) **For** all $t = 1 \ldots T$ and $i = 1 \ldots N$ **Determine** $\alpha_{t+1}(j)$ according to Equation 18.
6. **Let** $P(O|\Sigma, \lambda) = \sum_{i=1}^{N} \alpha_T(i)$
7. (second backward step) **For** all $t = T \ldots 1$ and $i = 1 \ldots N$ **Determine** $\beta_t(i)$ according to Equation 33 and determine $\gamma_t(i)$ according to Equation 50.

Table 2: Forward-Backward algorithm

1. **Input** Observation sequence $(O_1, \ldots, O_T)$ without labels, HMM λ.
2. **For all** $t = 1, \ldots, T$, $i = 1 \ldots, N$, **Let** $\tau_t(i) = 1$.
3. Enter the backward-forward-backward procedure (Table 1) at step 4. Only one forward and one backward iteration are thus required.

Note that it is possible but not necessary to adapt the $\delta_t(i)$ variable required to execute the Viterbi algorithm. The Viterbi algorithm is not required during training; it is used to find the most likely sequence of states of an already given HMM that corresponds to a new observation sequence. In contrast to the training data, the observation sequences encountered during the application phase will usually not contain any labels.

In order to adapt HMM parameters from a given set of partially labeled observation sequences, we need to calculate the variable $\xi_t(i,j)$ which quantifies the probability that a transition from S_i to S_j has occurred at time t. This variable is crucial to calculating the transition probability from S_i to S_j.

Lemma 5 (Computation of ξ) *Given a HMM with parameters λ, and observations $O = O_1, \ldots, O_T$ with corresponding possible states Σ, for any $1 \le t \le T$, $1 \le i \le N$, $1 \le j \le N$, we can compute $\xi_t(i,j) = P(q_t = S_i \wedge q_{t+1} = S_j \mid O, \Sigma, \lambda)$ as in Equation 57.*

$$\xi_t(i,j) = \begin{cases} \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \tau_{t+1}(j)}{P(O|\Sigma,\lambda)\tau_t(i)} & \text{if } i \in \sigma_t, j \in \sigma_{t+1} \\ 0 & \text{otherwise} \end{cases} \tag{57}$$

Note that $P(O \mid \Sigma, \lambda) = \sum_{i=1}^{N} \alpha_T(i)$.

Proof. First, we convert the conditional into a joint probability in Equation 59. In Equation 60 we first marginalize $P(O_1, \ldots, O_t, q_t = S_i|\Sigma, \lambda)$ (equals $\alpha_t(i)$), then $P(q_{t+1} = S_j|q_t = S_i, O_1, \ldots, O_t, \Sigma, \lambda)$, next $P(O_{t+2}, \ldots, O_T|q_t = S_i, q_{t+1} = S_j, O_1, \ldots, O_{t+1}, \Sigma, \lambda)$ (equals $\beta_{t+1}(j)$), and finally $P(O_{t+1}|q_t = S_i, q_{t+1} = S_j, O_1, \ldots, O_t, O_t, O_T, \Sigma, \lambda)$ (equals $b_j(O_{t+1})$ remains. Then, in Equation 62, we apply Bayes' equation. Note that $P(\sigma_{t+1}, \ldots, \sigma_T|q_{t+1} = S_j, q_t = S_i, \sigma_1, \ldots, \sigma_t, \lambda)$ equals zero if $j \notin \sigma_{t+1}$, it is undefined when $i \notin \sigma_t$; otherwise and under the Markov assumption it equals $P(\sigma_{t+2}, \ldots, \sigma_T|q_{t+1} = S_j, \lambda)$ which in turn equals $\tau_{t+1}(j)$. In Equation 63, we also refer to the Markov assumption to identify $P(q_{t+1} = S_j|q_t = S_i, \sigma_1, \ldots, \sigma_t, \lambda)$ as a_{ij}.

$$\begin{aligned}
\xi_t(i,j) &= P(q_t = S_i, q_{t+1} = S_j \mid O, \Sigma, \lambda) && (58)\\
&= \frac{P(q_t = S_i, q_{t+1} = S_j, O \mid \Sigma, \lambda)}{P(O|\Sigma,\lambda)} && (59)\\
&= P(O_{t+1}|q_t = S_i, q_{t+1} = S_j, O_1, \ldots, O_t, O_t, O_T, \Sigma, \lambda) \cdot \\
&\quad P(O_{t+2}, \ldots, O_T|q_t = S_i, q_{t+1} = S_j, O_1, \ldots, O_{t+1}, \Sigma, \lambda) \cdot \\
&\quad P(q_{t+1} = S_j|q_t = S_i, O_1, \ldots, O_t, \Sigma, \lambda) \frac{P(O_1, \ldots, O_t, q_t = S_i|\Sigma, \lambda)}{P(O|\Sigma,\lambda)} && (60)\\
&= \frac{\alpha_t(i) b_j(O_{t+1}) \beta_{t+1}(j)}{P(O|\Sigma,\lambda)} P(q_{t+1} = S_j|q_t = S_i, \sigma_1, \ldots, \sigma_T, \lambda) && (61)\\
&= \frac{\alpha_t(i) b_j(O_{t+1}) \beta_{t+1}(j)}{P(O|\Sigma,\lambda)} P(\sigma_{t+1}, \ldots, \sigma_T|q_{t+1} = S_j, q_t = S_i, \sigma_1, \ldots, \sigma_t, \lambda) \cdot \\
&\quad P(q_{t+1} = S_j|q_t = S_i, \sigma_1, \ldots, \sigma_t, \lambda) P(\sigma_{t+1}, \ldots, \sigma_T|q_t = S_i, \sigma_1, \ldots, \sigma_t, \lambda) && (62)\\
&= \begin{cases} \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \tau_{t+1}(j)}{P(O|\Sigma,\lambda)} & \text{if } i \in \sigma_t, j \in \sigma_{t+1} \\ 0 & \text{otherwise} \end{cases} && (63)
\end{aligned}$$

□

Given the above lemmata, we have the main building blocks for our new learning algorithm which works with partially labeled observations. Table 3 describes our Baum-Welch algorithm. Baum-Welch is an instantiation of the EM algorithm. In the E-step, the algorithm uses the current model λ to determine the probability $\gamma_t(i)$ of being in each state S_i at time step t

Table 3: Baum-Welch Algorithm for partially labeled observation sequences

1. **Input** Set $\mathcal{O}$ of m observation sequences $\mathcal{O} = \{(O_1^{(1)}, \ldots, O_{T_1}^{(1)}), \ldots, (O_1^{(m)}, \ldots, O_{T_m}^{(m)})\}$. Possible corresponding states $\{(\sigma_1^{(1)}, \ldots, \sigma_{T_s}^{(s)}\}$. Number of states N. Optionally an initial parameter set λ.
2. **If** no initial model λ is provided **Then** initialize (π, a, b) at random, but making sure that $\sum_{i=1}^{N} \pi_i = 1$, $\sum_o b_i(o) = 1$ (for all i), $\sum_{j=1}^{N} a_{ij} = 1$ (for all i), and that $0 < p < 1$ for all probabilities $p \in \{\pi, a, b\}$.
3. **Repeat**
   (a) (E-Step) **For** all $s = 1 \ldots m$ **Call** the backward-forward-backward procedure to calculate the $\tau_t^{(s)}(i)$, $\alpha_t^{(s)}(i)$, $\beta_t^{(s)}(i)$, and $\gamma_t^{(s)}(i)$.
   (b) **For** $s = 1 \ldots m$, $t = 1 \ldots T_s - 1$, $i \in \sigma_t$ and $j = 1, \ldots, N$, **Determine** $\xi_t^{(s)}(i, j)$ according to Equation 57.
   (c) (M-Step) **Let** $\pi_i = \frac{1}{m} \sum_{s=1}^{m} \gamma_1^{(s)}(i)$ (expected frequency of starting in state S_i).
   (d) **For** $i = 1 \ldots N$, $j = 1 \ldots N$ **Let** $a_{ij} = \frac{\frac{1}{m}\sum_{s=1}^{m}\sum_{t=1}^{T_s-1}\xi_t^{(s)}(i,j)}{\frac{1}{m}\sum_{s=1}^{m}\sum_{t=1}^{T_s-1}\gamma_t^{(s)}(i)}$ (expected number of transitions from S_i to S_j over number of transitions from S_i).
   (e) **For** $i = 1 \ldots N$ and all observation values o **Let** $b_i(o) = \frac{1}{m} \sum_{s=1}^{m} \frac{\sum_{t=1}^{T_s} \gamma_t^{(s)}(i) \cdot \delta(O_t^{(s)} = o)}{\sum_{t=1}^{T_s} \gamma_t^{(s)}}(i)$ (expected frequency of observing o when in state S_i; δ returns 1 if $O_t^{(s)} = o$, 0 otherwise).
4. **Until** the probabilities a_{ij} and $b_j(O_t)$ stay nearly constant over an iteration.
5. **Output** parameters $\lambda = (\pi, a, b)$.

(using the forward-backward procedure). In the M-step, the algorithm uses the calculated state probabilities to estimate the transition probabilities.

Our algorithm has two desirable properties. Like the regular Baum-Welch algorithm it converges to an (at least local) optimum in the parameter space. When the document is completely labeled, it reaches stability after the first iteration and behaves like the straightforward parameter estimation procedure for (not hidden) Markov models.

Theorem 1 (Correctness) *of the class from which a HMM is to be learned. Given observations $\mathcal{O} = \{O^{(s)} = O_1^{(s)}, \ldots, O_T^{(s)}\}$, partially labeled according to ℓ, the algorithm in Table 3 will converge to an estimated set of parameters λ that locally maximizes $P(\mathcal{O} \mid \ell, \lambda)$.*

Proof. Follows directly from the original proofs of Baum and Welch using our modified definition of ξ. □

4 Learning HMMs Actively

Unlabeled documents can be obtained very easily for most information extraction problems; only labeling tokens in a document imposes effort. An active learning approach to utilizing the available amount of user effort most effectively is to select, from the available unlabeled observations, the "difficult" ones of which to know the labels that would be most interesting.

When our objective is to minimize the probability of assigning a false state to an observation, then a Bayes-optimal extraction algorithm has to label each observation $O_t^{(s)}$ with the tag X_i that maximizes $P(S_i|O^{(s)}, \lambda) = \gamma_t^{(s)}(i)$ (or *none*, if $i > n$). We deviate from this optimal strategy when our parameter estimates λ' differ from the true parameters λ such that some state S_j seems to be most likely although state S_i really is most likely ($\max_i P(q_t = S_i|O, \lambda) \neq \max_i P(q_t = S_i|O, \lambda')$) We can see the difference between the probability of the most likely and that of the second most likely state as the confidence of the state given O. When such confidence values or margins can be defined for instances, then we often see empirically that instances with low margins are most relevant to adjust the hypothesis parameters (*e.g.,* [18, 4]).

In analogy, we define the *margin* $M(q_t|O, \lambda)$ of the observation at time t as the difference between the highest and second highest probability for a state (Equation 65).

$$
\begin{aligned}
M(q_t|O, \lambda) &= \max_i \{P(q_t = S_i|O, \lambda)\} - \max_{j \neq i} \{P(q_t = S_j|O, \lambda)\} && (64) \\
&= \max_i \{\gamma_t(i)\} - \max_{j \neq i} \{\gamma_t(j)\} && (65)
\end{aligned}
$$

Intuitively, the margin can be seen as quantifying how "difficult" (low margin) or "easy" (large margin) an example is. Our active HMM learning algorithm (Table 4) first learns an initial model λ_1 from a set of partially labeled observation sequences. It then determines the margins of all observations and starts asking the user to label those observations which have a particularly low margin. The Baum-Welch algorithm is restarted, using the previous parameters λ_{k-1} as initial model and adapting λ_k to the new data.

Table 4: Active Revision of hidden Markov Models

1. **Input** Set $\mathcal{O}$ of m Sequences $\mathcal{O} = \{(O_1^{(1)}, \ldots, O_{T_1}^{(1)}), \ldots, (O_1^{(m)}, \ldots, O_{T_m}^{(m)})\}$ of observations; Labeling function $l : O_t^{(s)} \mapsto \{S_1, \ldots, S_n, unknown, nolabel\}$; Number of states N; query parameter n_q.

2. **Call** the Baum-Welch algorithm to determine initial parameters λ_1.

3. **For** $k = 2 \ldots \infty$ **Repeat**

   (a) **For** $s = 1 \ldots m$ **Call** the forward-backward algorithm.

   (b) **For** $s = 1 \ldots m$ and $t = 1 \ldots T_s$ **Let** $M(q_t|O^{(s)}) = \max_i\{\gamma_t(i)\} - \max_{j \neq i}\{\gamma_t(j)\}$.

   (c) **Ask** the user to label the n_q unlabeled observation $O_t^{(s)}$ ($l(O_t^{(s)}) = nolabel$) with smallest margin $M(q_t|O^{(s)})$. Update the labeling function l.

   (d) **Estimate** the new model parameters λ_k using the Baum-Welch algorithm with initial model λ_{k-1}.

4. **Until** the user gets tired of labeling data.

5. **Return** HMM parameters $\lambda_i = (\pi, a, b)$.

5 Experiments

For our experiments, we generated HMMs with variable numbers of background and target states at random. We used these HMMs to generate unlabeled observation sequences; we label a number of initial observations drawn at random. We then study how the error developes with the number of additional labels added to the observation sequences according to three strategies. The "random" stragegy is to label randomly drawn observations; the "margin" strategy is to label "difficult" observations with smallest margins; this corresponds to our active hidden Markov model. As a control strategy ("large margins"), we also try selecting "easy" observations that have the largest margins. If "margins" is really better than "random", we expect "large margins" to perform worse.

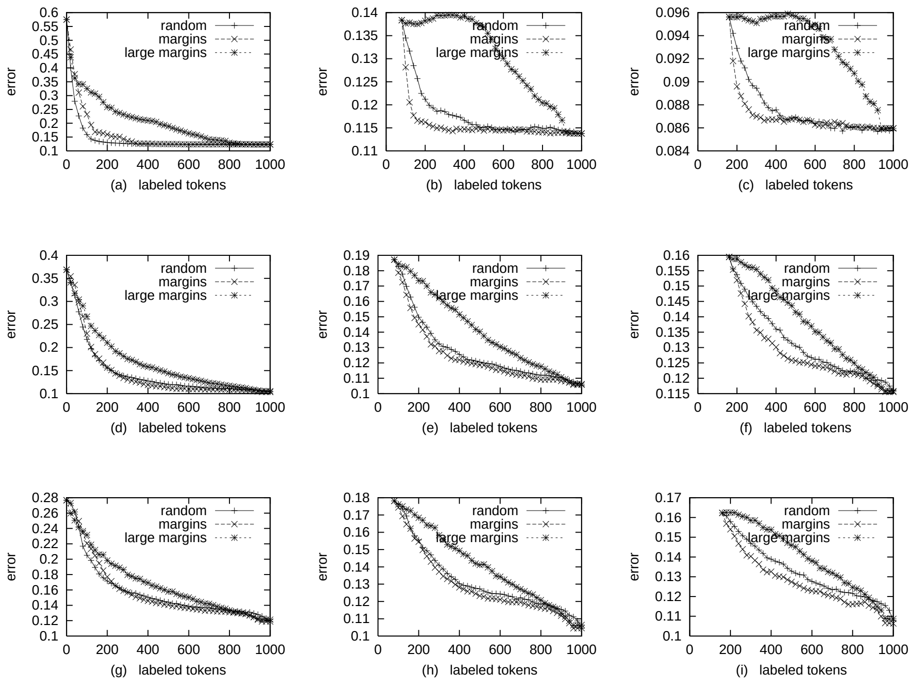

Figure 1: Error rate of active and regular learning over number of labeled observations. (a)-(c), easy HMM; initial sample contains (a) no (b) 80 (c) 160 labeled observations drawn at random. (d)-(f) medium size HMM; initial sample with (d) no (e) 80 (f) 160 initial labels. (g)-(i) large HMM; (g) no (h) 80 (i) 160 initial labels.

We used three different HMM sizes. The "easy" HMM consists of one background and two target states. Each state emits three out of 20 observations with randomly drawn probabilities. We generated 50 sequences of 20 initially unlabeled observations. The "medium size" HMM possesses 10 nodes, and the "large" HMM consists of 15 states. The curves in Figure 1 are averages over 50 leaning problems. The initial sample contains only unlabeled observations (Figures 1a for the easy, 1d for the medium size and 1g for the hard learning problem), labels of 80 (Figures 1b, 1e, and 1h, respectively) and 160 observations (Figures 1c, 1f, and 1i) drawn at random.

In Figures 1a, 1d, and 1g we see a slight but significant advantage of random selection of observations over selecting observations with small margins. Using only difficult observations from the beginning is not beneficial on average. In the later phase of learning, observation selection by small margins gains a small advantage. The benefit of the margin strategy becomes more clearly visible, when the initial sample contains the labels of 80 (Figures 1b, 1e, and 1h) or 160 (Figures 1c, 1f, and 1i) observations drawn at random, and only from then on observations with smallest margins are selected. For the small HMM learning problem (when much unlabeled data relative to the problem complexity is available), the bottom line error is reached after about 300 labels under the margin strategy and after 600 labeled observations under the random strategy.

Using active learning with small margin examples after 70 initial random observations seems to be most beneficial. In this case (Figure 1b), the base level error is reached after less than 200 examples for active and after 1000 examples regular learning – *i.e.,* five times fewer labels are needed. Choosing only the most "easy" examples (large margins) is clearly a bad strategy in all cases. Our experiments show that, at least for the classes of HMM that we generated, using only difficult low-margin observations for learning from the beginning results in higher error rates. However, when the training is started by labeling randomly drawn observations and the active HMM chooses difficult low-margin observations after that initial phase, then a significant improvement is achiever over regular HMMs that can result in the sufficiency of many times fewer labeled examples.

6 Discussion and Related Work

An alternative, conditional hidden Markov model for information extraction has been described by [14]. Instead of using the usual distributions a_{ij} and $b_i(O_t)$, the alternative model uses a conditional probability $P(q_{t+1}|q_t, O_{t+1})$. The conditional model is not generative any more – *i.e.,* it is not possible to determine $P(O|\lambda)$, or to draw observation sequences according to some given λ.

In the generative model a_{ij} and $b_i(O_t)$ have to be estimated. That is $|Q|^2 + |Q| \times |O|$ probabilities. By contrast, in the conditional model, $|Q|^2 \times |O|$ probabilities have to be estimated. Hence, fewer probabilities need to be estimated in the generative model but, in the conditional model it may be possible to model some processes using fewer states.

Given a set of models $\{\lambda_i\}$ (*e.g.,* describing different categories of web pages) and an observation sequence O, in the generative model we can calculate the $P(O|\lambda_i)$ and thus use HMMs for classification tasks. This idea seems promising because we can thus use information that lies beyond the usual bag-of-words representation for text classification. This probability cannot be determined in the conditional model.

Furthermore, it is not clear whether Viterbi and Baum-Welsh algorithms exist in the conditional model, or whether they can only be approximated. Consider, for instance, the definition of the forward variable α in the conditional model (Equation 66). We can factorize q_t as in Equation 67. The Markov assumption then takes us to (68). Unfortunately, (68) is, in general, not equal to (69) which is equal to the recursive algorithm that [14] propose for computation of the α. Equation 68 is only equal to Equation 69 if $P(q_t = S_i|O_1, \ldots, O_t, O_{t+1}) = P(q_t = S_i|O_1, \ldots, O_t)$ which is not covered by the Markov assumption.

$$\alpha_{t+1}(j) = P(q_{t+1} = S_j|O_1, \ldots, O_{t+1}) \tag{66}$$

$$= \sum_{i=1}^{N} P(q_{t+1} = S_j | q_t = S_i, O_1, \ldots, O_{t+1}) P(q_t = S_i | O_1, \ldots, O_{t+1}) \quad (67)$$

$$= \sum_{i=1}^{N} P(q_{t+1} = S_j | q_t = S_i, O_{t+1}) P(q_t = S_i | O_1, \ldots, O_t, O_{t+1}) \quad (68)$$

$$\neq \sum_{i=1}^{N} P(q_{t+1} = S_j | q_t = S_i, O_{t+1}) P(q_t = S_i | O_1, \ldots, O_t) \quad (69)$$

$$= \sum_{i=1}^{N} P(q_{t+1} = S_j | q_t = S_i, O_{t+1}) \alpha_t(i) \quad (70)$$

A different way of seeing the derivation is the following interpretation in which the $\alpha_t(i)$ are conditioned on the entire observation sequence.

$$\alpha_{t+1}(j) = P(q_{t+1} = S_j | O) \quad (71)$$

$$= \sum_{i=1}^{N} P(q_{t+1} = S_j | q_t = S_i, O) P(q_t = S_i | O) \quad (72)$$

$$= \sum_{i=1}^{N} P(q_{t+1} = S_j | q_t = S_i, O) \alpha_t(i) \quad (73)$$

In this interpretation, the problem occurs only one step later. We now have to estimate $P(q_{t+1} = S_j | q_t = S_i, O)$ and, in order to make this problem tractable, a small time window focused around O_t is considered.

$$P(q_{new} = S_j | q_{old} = S_i, O) \quad := \quad P(q_{new} = S_j | q_{old} = S_i, O_{t-k}, \ldots, O_{t+k}) \quad (74)$$

Note that we now have a set of transition probabilities that are time dependent! We now have to estimate an array of $P(q_{new} | q_{old}, O)$ for each time step t – or artificially assume that the transition probability is constant. But this assumption is not covered by the Markov assumption and not consistent with the update formula. This "bug" in the derivation of the conditional α variable is reflected by the unsatisfactory empirical performance of conditional MEMMs [13]. γ.

Conditional random fields are are also a promising approach to avoid the problems associated with conditional Markov models. One problem of conditional random fields if the slow convergence of the Improved Iterative Scaling algorithms. Empirical comparisons of HMMs and CRFs would be interesting.

An alternative heuristic for selecting observation sequences (not, as in our approach, observations) for labeling has been presented by [16].

A considerable restriction of the model discussed here is that the tagging takes place on a token level – *i.e.,* it is not possible to enclose a token *sequence* in (nested) tags. Only when all tokens are labeled with a tag that corresponds to exactly one state, then cascaded Markov models [3] solve this problem. For the more general case of partially labeled documents, the hierarchical hidden Markov model [7] has to be adapted to partially labeled token sequences, analogously to our adaptation of the standard hidden Markov model.

Acknowledgment

We would like to thank Christian Decomain, Borislav Popov, Damyan Ognianov, and Atanas Kiryakov for their contributions.

References

[1] P. Baldi, Y. Chauvin, Y. Hunkapliier, and M. McClure. Hidden markov models of biological primary sequence information. *Proceedings of the National Academy of Sciences*, 91(3):1059–1063, 1994.

[2] T. Berners-Lee. Semantic web road map. Internal note, World Wide Web Consortium, 1998.

[3] T. Brants. Cascaded markov models. In *Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics*, 1999.

[4] D. Cohn, Z. Ghahramani, and M. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[5] Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew K. McCallum, Tom M. Mitchell, Kamal Nigam, and Seán Slattery. Learning to construct knowledge bases from the World Wide Web. *Artificial Intelligence*, 118(1-2):69–113, 2000.

[6] L. Eikvil. Information extraction from the world wide web: a survey. Technical Report 945, Norwegian Computing Center, 1999.

[7] S. Fine, Y. Singer, and N. Tishby. The hierarchical hidden markov model: Analysis and applications. *Machine Learning*, 32:41–64, 1998.

[8] G. Grieser, K. Jantke, S. Lange, and B. Thomas. A unifying approach to html wrapper representation and learning. In *Proceedings of the Third International Conference on Discovery Science*, 2000.

[9] Ralph Grishman and Beth Sundheim. Message understanding conference - 6: A brief history. In *Proceedings of the International Conference on Computational Linguistics*, 1996.

[10] N. Hsu and M. Dung. Generating finite-state transducers for semistructured data extraction from the web. *Journal of Information Systems, Special Issue on Semistructured Data*, 23(8), 1998.

[11] B. Juang and L. Rabiner. Hidden markov models for speech recognition. *Technometrics*, 33:251–272, 1991.

[12] N. Kushmerick. Wrapper induction: efficiency and expressiveness. *Artificial Intelligence*, 118:15–68, 2000.

[13] John Lafferty, Fernando Pereira, and Andrew McCallum. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*, 2001.

[14] Andrew McCallum, Dayne Freitag, and Fernando Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000.

[15] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285, 1989.

[16] Nick Roy and Andrew McCallum. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the International Conference on Machine Learning*, 2001.

[17] Kristie Seymore, Andrew McCallum, and Roni Rosenfeld. Learning hidden markov model structure for information extraction. In *AAAI'99 Workshop on Machine Learning for Information Extraction*, 1999.

[18] V. Vapnik. *Statistical Learning Theory.* Wiley, 1998.