*Künstliche Intelligenz* 2/2002

# Learning Hidden Markov Models for Information Extraction Actively from Partially Labeled Text

Tobias Scheffer[1], Stefan Wrobel[1], Borislav Popov[2],
Damyan Ognianov[2], Christian Decomain[1], Susanne Hoche[1]
[1] University of Magdeburg, FIN/IWS, PO Box 4120, 39016 Magdeburg, Germany
[2] Sirma AI, Ontotext Lab, 8 A Christo Botev Blvd. 1000 Sofia, Bulgaria
{scheffer, wrobel, decomain, hoche}@iws.cs.uni-magdeburg.de,
{borislav, damyan}@sirma.bg

May 25, 2002

**Abstract**

A vast range of information is expressed in unstructured or semi-structured text, in a form that is hard to decipher automatically. Consequently, it is of enormous importance to construct tools that allow users to extract information from textual documents as easily as it can be extracted from structured databases. Information Extraction (IE) systems identify predetermined relevant information in text documents from some specific domain and fill it into a structured form. Our approach to solving this problem is based on Hidden Markov models (HMMs) which can be generated automatically from manually labeled example documents. We consider the challenging task of learning HMMs when only *partially (sparsely) labeled* documents are available for training. In order to further reduce the amount of data labeling effort a user has to invest, we describe how our algorithm can be naturally extended to 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 labeling 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 [5, 3], 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. While first IE algorithms were hand-crafted sets of rules (*e.g.,* [6]), researchers soon turned to learning extraction rules from hand-labeled documents (*e.g.,* [8, 10]). 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) [11] for this purpose.

Markov model algorithms that are used for part-of-speech tagging [1], as well as known hidden Markov models for IE [11] require the training documents to be labeled *completely*, *i.e.,* each token is manually given an appropriate label. Clearly, this is an expensive process. We therefore concentrate on the task of learning information extraction models, in particular hidden Markov models, from *partially* labeled texts, and develop appropriate EM-style algorithms.

As we will show, by using additional *unlabeled* documents as they are usually readily available in most applications, we can perform *active learning* of HMMs. The idea of *active learning* algorithms (*e.g.,* [2]) is to identify "difficult" unlabeled observations and ask the user to label them. Such algorithms are known for classification (*e.g.,* [2]), clustering [7], and regression [9]; here, we present an algorithm for active learning of hidden Markov models.

The paper is organized as follows. We give account of the IE task and loss functions in Section 2, followed by a description of hidden Markov models (Section 3). We then discuss how a given HMM can be applied to an IE problem (Section 4) and how HMMs can be learned from example documents (Section 5). We discuss our active learning approach (Section 6), and our empirical results (Section 7); Section 8 concludes.

## 2 Information Extraction

IE systems identify predetermined relevant information in text documents from some specific domain, extract it and convert it into a structured format such as a database or an XML document.

As an example, consider the problem of processing flight confirmations sent to an airline by email automatically. In order to identify the flight database record to be confirmed, details like flight numbers, outbound and inbound dates, as well as names of passengers have to be extracted from an unstructured text.

Another interesting application is in the domain of computational biology. A great amount of biological data collections are maintained as internet repositories. These data collections can be queried over the Internet. The answers are generally provided in form of HTML-pages. In order to further process pertinent information from such HTML documents, it is necessary to identify, for example, enzyme functional data, reaction type, inhibitors and molecular weight of queried enzymes in this semi-structured HTML output.

Let us give a definition of the task considered in this paper. A document is a sequence of observations, $O = (O_1, \ldots, O_T)$. An observation $O_t$ corresponds to a token of the document. A token is a vector of attributes generated by a collection of NLP tools. Attributes may include the word stem, part of speech, HTML context, and many other properties of the word, sentence, or paragraph.

The IE task is to attach a semantic *tag* $X_t$ to a token $O_t$; observations can also be left untagged (special tag *none*). An extraction algorithm or function $f$ maps an observation sequence $O_1, \ldots, O_T$ to a single sequence of tags $(X_1, \ldots, X_T), X_t \in \{x_1, \ldots, x_n, none\}$ (multi-valued assignments would have to be handled by using several IE models, one per label).

An IE problem is then given by an (unknown) joint distribution $P(X_1, \ldots, X_T, O_1, \ldots, O_T)$ over tags and observations. Example documents are assumed to have been drawn according to the resulting marginal distribution $P(O_1, \ldots, O_T)$ over documents; the *IE learning problem* is to find an extraction function $f$ (given a set of example documents) that minimizes a chosen error criterion.

Depending on the application, there are several ways to define the error criterion to be minimized. In many applications, costs may arise for each false tag assigned to a token. In such cases, we can define the *per-token error* $E_{token}$ as the probability of tokens with false tags.

*Precision* and *recall* are other popular error criteria that can be defined for problems for which only one tag ($X_t \in \{x, none\}$) is available. Precision refers to the amount of correct tags $x$ assigned by $f$ relative to the number of all (correctly and incorrectly) assigned tags $x$. Recall reflects the amount of tags $x$ correctly assigned by $f$ relative to the total amount of tags $x$ that should have been assigned.

## 3 Hidden Markov Models

Hidden Markov models (HMMs) (see [12], for an introduction) are a very robust statistical method for structural analysis of sequential data. HMMs are finite state machines with stochastic state and observation sequences.

A HMM $\lambda = (\pi, a, b)$ consists of finitely many states $\{S_1, \ldots, S_N\}$. The state that the HMM is in at time $t$ is denoted as $q_t$ ($q_t$ is the random variable, $S_1, \ldots, S_N$ its possible values). At each time step $t$, the HMM generates an observation $O_t$. The array $\pi_i = P(q_1 = S_i|\lambda)$ denotes the probabilities of starting in state $S_i$; $a_{ij} = P(q_{t+1} = S_j|q_t = S_i, \lambda)$ quantifies the probability of a transition from state $S_i$ to $S_j$. Each state is characterized by a probability distribution $b_i(O_t) = P(O_t|q_t = S_i, \lambda)$ over the observations. At each time step $t$, the HMM changes its state according to $a$ and an observation is generated according to $b$.

The states through which the model passes are invisible to the outside, only the observation sequence which is a probabilistic function of the state sequence is known; hence the name hidden Markov model. HMMs are useful to model how underlying events probabilistically induce surface events and thus have been found suitable to model natural language.

In this context, an HMM with parameters $\lambda$ can be seen as a stochastic "grammar" of a focused class of documents which describes probabilistically which sequences of words (*i.e.,* documents) occur in this class. Observations are the individual words of a document, and the HMM state in which a word has been emitted represents its semantic role, or meaning.

An HMM for information extraction possesses one state for each of the tags which we want to attach to words, and it possesses a number of "background" states representing the meaning of those words which we are not interested in extracting. We denote the target states as $S_1, \ldots, S_n$ and the corresponding tags $x_1, \ldots, x_n$. The background states which represent information we are not interested in are denoted $S_{n+1}, \ldots, S_N$.

Underlying HMMs is the Markov assumption which says that each state $q_t$ conditionally only depends on its direct predecessor (higher-order HMMs generalize this to $k$ preceeding states) – $P(q_{t+1}|q_1, \ldots, q_t) = P(q_{t+1}|q_t)$. In other words, the states $q_t$ summarize all past information relevant for $q_{t+1}$.

# 4 Applying HMMs

Suppose that we have a Hidden Markov model $\lambda$ which describes a class of documents which we are interested in, and we have a new document $(O_1, \ldots, O_T)$ which is an instance of this class. For each word $O_t$, we want to identify the hidden state $S_i$, and thereby the meaning, corresponding to it. Once we have identified the states $q_t$, we simply have to extract those tokens which correspond to one of the target states $S_1, \ldots, S_n$, and label them with the corresponding tags $x_1, \ldots, x_n$.

The most important step in this process of applying a HMM $\lambda$ is to determine the probabilities $P(q_t = S_i|O_1, \ldots, O_T, \lambda)$ that the HMM is in state $S_i$ at time $t$. This probability is usually abbreviated $\gamma_t(i)$. Once we have determined the $\gamma_t(i)$, the optimal extraction strategy depends on the precise optimization criterion. In order to minimize the number of false labels, we have to assign tag $x_i$ to $O_t$ if state $S_i$ is most likely ($i$ maximizes $\gamma_t(i)$). A certain precision/recall balance is achieved by assigning tag $x_i$ to a word if $\gamma_t(i)$ exceeds a threshold $\theta$. This threshold $\theta$ then balances precision against recall.

Let us now focus on the problem of calculating $\gamma_t(i) = P(q_t = S_i|O_1, \ldots, O_T, \lambda)$. This problem is a special case of a slightly more general problem which we also have to solve in order to find a learning algorithm for HMMs in the next section. In the more general case, some tokens may possess labels that the user has attached. This immediately restricts the state of labeled tokens (to all states consistent with user-assigned label or labels), but the user-defined labels can also have an impact on nearby unlabeled words. We denote the set of all state indices which are consistent with the user-defined label for word $O_t$ as $\sigma_t$. When we apply a given HMM to a new document, then no labels are present and hence $\sigma_t = \{1, \ldots, N\}$ which means that all states are possible for each word.

Our solution is inspired by the message passing algorithm for Bayesian networks; it extends the standard message passing algorithm by taking the constraints $\sigma_t$ into account. In the light of our solution, the close relation of the forward-backward algorithm for hidden Markov models and the message passing algorithm for Bayesian networks can be recognized. We decompose $\gamma_t(i) = P(X|E_X^-, E_X^+)$ into $\alpha_t(i) = P(X|E_X^+)$, and $\beta_t(i) = P(E_X^-|X, E_X^+)$, where $E_X^-$ is the evidential support and $E_X^+$ is the causal support.

$$\gamma_t(i) = P(q_t = S_i|\sigma_1, \ldots, \sigma_T, O_1, \ldots, O_T, \lambda) \tag{1}$$

$$= \underbrace{c}_{normalization} \underbrace{P(q_t = S_i|O_1, \ldots O_{t-1}, \sigma_1, \ldots, \sigma_{t-1}, \lambda)}_{\alpha_t(i)} \underbrace{P(O_t, \ldots, O_T, \sigma_t, \ldots, \sigma_T|q_t = S_i, \lambda)}_{\beta_t(i)} \tag{2}$$

Please note that in order to work for our extended problem, the $\alpha$ and $\beta$ defined above have been chosen differently from the corresponding probabilities used in the classical HMM algorithm as in [12]. Straightforward recursive evaluation of $\alpha_t(i)$ and $\beta_t(i)$ would be prohibitively expensive. In Theorems 1 and 2 we present a dynamic programming algorithm which calculates the $\alpha_t(i)$ and $\beta_t(i)$ efficiently.

**Definition 1** *We define $\alpha_t(i)$ as the probability of arriving in state $S_i$ at time $t$ given an initial observation sequence and labels (Equation 3); $\beta_t(i)$ is the probability of encountering the remaining observations and* *labels given the state at time t (Equation 4). Finally, $\gamma_t(i)$ is the probability of state $S_i$ at time t given a partially labeled observation sequence.*

$$
\begin{aligned}
\alpha_t(i) &= P(q_t = S_i | O_1, \ldots O_{t-1}, \sigma_1, \ldots, \sigma_{t-1}, \lambda) && (3)\\
\beta_t(i) &= P(O_t, \ldots, O_T, \sigma_t, \ldots, \sigma_T | q_t = S_i, \lambda) && (4)\\
\gamma_t(i) &= P(q_t = S_i | \sigma_1, \ldots, \sigma_T, O_1, \ldots, O_T, \lambda) && (5)
\end{aligned}
$$

For the proofs of the following Theorems, we refer the reader to [13].

**Theorem 1** $\alpha_t(i)$ *can be computed recursively as in Equations 6 and 7.*

$$
\begin{aligned}
\alpha_1(i) &= \pi_i && (6)\\
\alpha_{t+1}(j) &= \frac{\sum_{i \in \sigma_t} a_{ij} b_i(O_t) \alpha_t(i)}{\sum_{k \in \sigma_t} b_k(O_t) \alpha_k(i)} && (7)
\end{aligned}
$$

**Theorem 2** $\beta_t(i)$ *can be computed recursively as in Equations 8 and 9.*

$$
\begin{aligned}
\beta_T(i) &= \begin{cases} b_i(O_T) & \text{if } q_T \in \sigma_T \\ 0 & \text{otherwise} \end{cases} && (8)\\
\beta_t(i) &= \begin{cases} b_i(O_t) \sum_{j \in \sigma_{t+1}} a_{ij} \beta_{t+1}(j) & \text{if } q_t \in \sigma_t \\ 0 & \text{otherwise} \end{cases} && (9)
\end{aligned}
$$

**Theorem 3** $\gamma_t(i)$ *can be computed as in Equation 10.*

$$
\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j \in \sigma_t} \alpha_t(j)\beta_t(j)} \qquad (10)
$$

Theorems 1, 2, and 3 say that it is possible to determine $\gamma_t(i)$ in linear time. The forward-backward algorithm achieves this as follows.

1. **For** $t = 1 \ldots T$, calculate the $\alpha_t(i)$ according to Theorem 1.
2. **For** $t = T \ldots t$, calculate the $\beta_t(i)$ according to Theorem 2 and, in the same pass, calculate the $\gamma_t(i)$ according to Theorem 3.

# 5 Learning HMMs from Partially Labeled Data

We have now seen how we can use a given HMM for information extraction, but we still have to study how we can learn a HMM from a set of example sequences. We assume that a set of initially unlabeled documents is available and the user then labels some of the *words* (not whole sequences) with the appropriate tag.

We express labels in terms of sequences of possible states $\sigma_1, \ldots, \sigma_T$, $\sigma_t \subseteq \{1, \ldots, N\}$. The user may label a token with a tag $x_i$ which means that the HMM must have been in state $S_i$ while emitting it; hence, $\sigma_t = \{i\}$. Tokens may be labeled as not possessing a tag which implies that the HMM must have been in one of the background states ($\sigma_t = \{n+1, \ldots, N\}$); or may be left unlabeled which says nothing about the state ($\sigma_t = \{1, \ldots, N\}$). This extends the setting of the part-of-speech tagging problem in which each token has a label that corresponds to exactly one state.

The *Baum-Welch algorithm* (see, for example, [12]) can be used to estimate the most likely model parameters $\lambda$ given a set of observation sequences. The principle difficulty which this algorithm addresses is that, in order to determine the transition and emission probabilities, the states that correspond to the observations would have to be known. Unfortunately, the states of unlabeled tokens are hidden (up to the constraints imposed by the labels). In the so-called *E-step*, the algorithm calculates, for each token, the probability distribution over the states based on the current estimate of the transition and emission probabilities, taking the labels into account. Based on these distributions over states, in the so-called *M-step* we update the emission and transition probabilities ("EM-algorithm").

1. Start with a random model $\lambda$.
2. Use the forward-backward algorithm to estimate the state probabilities $\gamma_t(i)$ based on the current model.
3. Based on the calculated state probabilities $\gamma_t(i)$, estimate the transition probabilities $a_{ij}$ by counting how many transitions from each $S_i$ to $S_j$ occur according to $\gamma_t(i)$.
4. Also, calculate, for each state $S_i$, how frequently the observations $O$ are observed in that state. Thus, estimate $b_i(O_t)$.
5. Recur from step 2 until the parameters $\lambda$ stay nearly constant over an iteration.

While the Baum-Welch algorithm itself is well-known, our contribution here is to present an instantiation for information extraction that handles partially labeled observation sequences in a mathematically sound way. This is made possible by Theorems 1 through 3 which say that we can efficiently calculate a modified $\gamma_t(i)$ variable that takes all (including non-local) effects of labels on the state probabilities into account.

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.

# 6 Active Revision of HMMs

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 tokens, those which are potentially most interesting.

If our objective is to minimize the number of false tags, a Bayes-optimal extraction algorithm has to label each token $O_t$ with the tag $X_i$ that maximizes $P(q_t = S_i|O, \lambda) = \gamma_t(i)$ (or *none*, if $i > n$), where $O = O_1, \ldots, O_T$ is a given observation sequence. We deviate from this optimal strategy when our parameter estimates $\lambda'$ differ from the true parameters $\lambda$ 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, or *margin* of the state given $O$.

We define the *margin* $M(q_t|O, \lambda)$ of the token that we read at time $t$ as the difference between the highest and second highest probability for a state (Equation 12):

$$\begin{aligned} M(q_t|O,\Sigma,\lambda) &= \max_i\{P(q_t = S_i|O,\Sigma,\lambda)\} \qquad (11) \\ &\quad - \max_{j\neq i}\{P(q_t = S_j|O,\Sigma,\lambda)\} \\ &= \max_i\{\gamma'_t(i)\} - \max_{j\neq i}\{\gamma'_t(j)\} \qquad (12) \end{aligned}$$

Intuitively, the margin can be seen as quantifying how "difficult" (low margin) or "easy" (large margin) a token is. Our active HMM learning algorithm first learns an initial model $\lambda_1$ from a set of partially labeled documents. It then determines the margins of all tokens and starts asking the user to label those tokens which have a particularly low margin. The Baum-Welch algorithm is restarted, using the previous parameters $\lambda_{k-1}$ as initial model and adapting $\lambda_k$ to the new data.

# 7 Experiments

To empirically test our active learning algorithm, we performed two sets of experiments, one with synthetic artificial data, one with data from a real-life problem (flight confirmation e-mails).

For the synthetic experiments, we generated HMMs with variable numbers of background and target states at random. We used these HMMs to generate unlabeled observation sequences; we labeled a number of initial tokens drawn at random. We then studied how the error develops with the number of additional labels added to the observation sequences according to three strategies. The first is our active learning algorithm proposed in this paper, *i.e.,* asking the user for labels on observations with smallest margins; we refer to our approach simply as the "margin" strategy. By contrast, the "random" strategy is to label randomly drawn observations, and the "large margins" strategy is the opposite to our proposed strategy, *i.e.,* selecting tokens that have the *largest* margins. If our proposed strategy ("margins") is really better than "random", we expect "large margins" to perform worse.

We used three different HMM sizes. The "easy" hidden Markov model 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; each state has nonzero transition probabilities to five other states. The curves in Figure 1 are averages over 50 learning problems.

The initial sample contains only unlabeled tokens (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 tokens (Figures 1c, 1f, and 1i) drawn at random.

In Figures 1a, 1d, and 1g, we see a slight but significant advantage of random token selection over selecting tokens with small margins. Using only difficult tokens from the beginning is not beneficial on average. In the later phase of learning, token 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) tokens drawn at random, and only from then on tokens 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 tokens under the random strategy.

Using active learning with small margin examples after 70 initial random tokens 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 for 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 tokens for learning from the beginning results in higher error rates. However, when the training is started by labeling randomly drawn tokens and the active HMM chooses difficult low-margin tokens after that initial phase, then a significant improvement is achieved over regular HMMs that can result in the sufficiency of many times fewer labeled examples.

For our real-life experiments, the task was to analyze flight confirmation emails written to an airline. In order to process flight confirmations automatically, it is necessary to identify passenger names, flight numbers, outbound and inbound dates, and locations. This problem is very hard (even for a human customer support agent) because most emails are very unstructured, ungrammatical, and often imprecise.

Figure 2 shows the precision and recall curves (depending on the number of example documents). The figure shows that reasonably small numbers of documents (with a total labeling effort of a few hours) suffice to obtain precision and recall rates of between 60 and 80%.

## 8 Discussion and Related Work

The range of possible problems which can be addressed effectively with a robust, generic information extraction tool is very wide, ranging from automatically generating databases for price comparisons to corporate intelligence.

We have discussed how the hidden Markov model framework can be adapted to deal with sparsely labeled observation sequences, as they naturally occur in the information extraction problem. We also proposed a strategy for active selection of unlabeled "difficult" tokens for labeling. Our experiments show that active data selection can reduce the data labeling effort substantially.

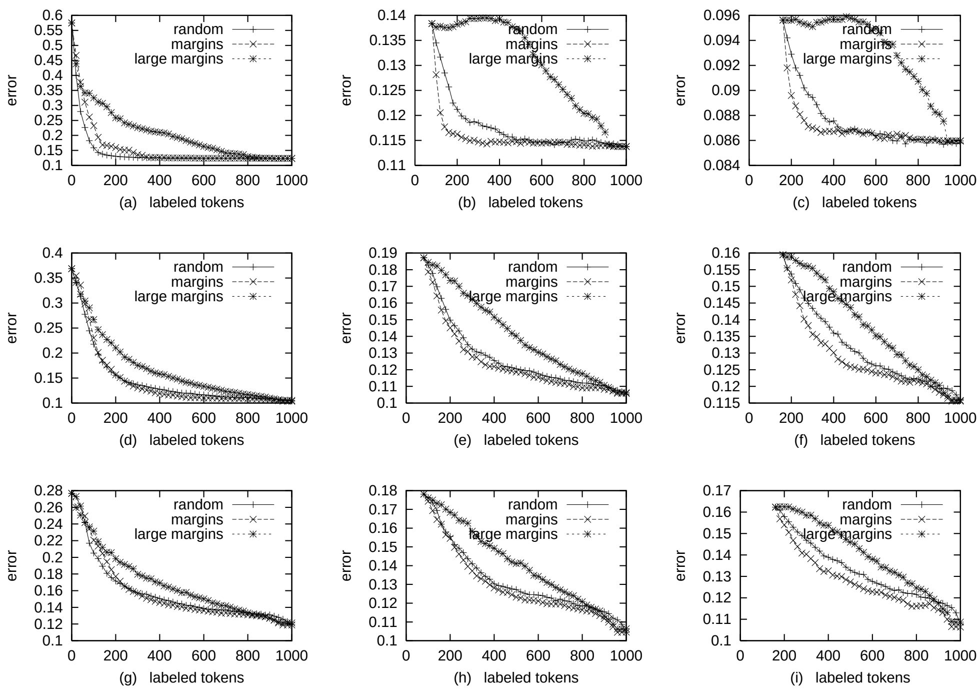


Figure 1: Error rate of active and regular learning over number of labeled tokens. (a)-(c), easy HMM; initial sample contains (a) no (b) 80 (c) 160 labeled tokens drawn at random. (d)-(f) medium size HMM; initial sample with (d) no (e) 80 (f) 160 initial labels. (g)-(i) large HMM; (g) no (h) 80 (i) 160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

An alternative non-generative hidden Markov model for information extraction has been described in [11]. Instead of using the usual distributions $a_{ij}$ and $b_i(O_t)$, the alternative model uses a conditional probability $P(q_{t+1}|q_t, O_{t+1})$. Consequently, the non-generative model possesses $|Q|^2 \times |O|$ probabilities which have to be estimated, in contrast to $|Q|^2 + |Q| \times |O|$ probabilities in the generative model.

Given a set of models $\{\lambda_i\}$ (*e.g.,* describing different categories of web pages) and an observation sequence $O$, we can calculate the $P(O|\lambda_i)$ in our generative model and thus use HMMs for text 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 non-generative model. Furthermore, the Forward-Backward and Baum-Welch algorithm in the non-generative model are based on some assumptions which we believe are difficult to motivate; see [14] for a detailed discussion.

Durbin et al. [4] have proposed a modification of the Baum-Welch algorithm for partially labeled observation sequences in which probabilities for events that are inconsistent with the labels are simply set to zero and calculated as if no labels were present otherwise. Although this procedure is simple and may in some cases lead to the desired results, it deviates from our mathematically strict approach of calculating the state probability *given* the labels.

## Acknowledgements

Part of this work was supported by DFG (German Science Foundation) grant FOR345/1-1TP6 as part of the project cluster “Information Fusion”.

## References

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

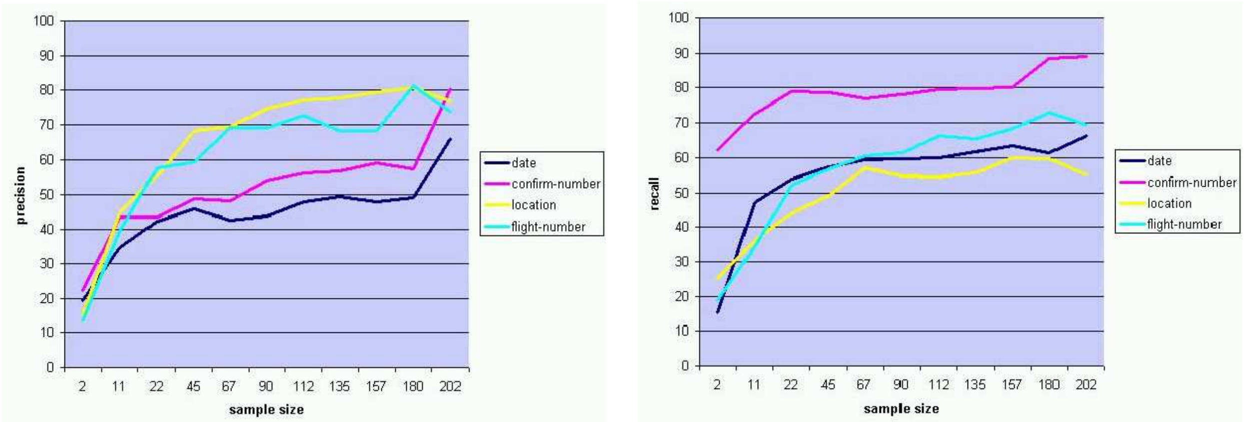


Figure 2: Precision and recall rates obtained for extraction of flight details from customer emails depending on the number of examples.

[2] D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

[3] 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.

[4] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis: Probabilistic models of proteins and nucleic acids.* Cambridge University Press, 1998.

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

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

[7] Thomas Hofmann and Joachim M. Buhmann. Active data clustering. In *Advances in Neural Information Processing Systems*, volume 10, 1998.

[8] 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.

[9] Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. In *Advances in Neural Information Processing Systems*, volume 7, pages 231–238, 1995.

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

[11] 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.

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

[13] T. Scheffer, C. Decomain, S. Hoche, D. Ognianov, B. Popov, and S. Wrobel. Learning hidden markov models for information extraction actively from partially labeled text. Technical report, University of Magdeburg, 2001.

[14] Tobias Scheffer and Stefan Wrobel. Active learning of partially hidden markov models. In *Proceedings of the ECML/PKDD Workshop on Instance Selection*, 2001.