Mining the Web With Active Hidden Markov Models*

Tobias Scheffer[1,2], Christian Decomain[1,2], and Stefan Wrobel[1]
[1] University of Magdeburg, P.O. Box 4120, 39016 Magdeburg, Germany
[2] SemanticEdge, Kaiserin-Augusta-Allee 10-11, 10553 Berlin, Germany
{scheffer, decomain, wrobel}@iws.cs.uni-magdeburg.de

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. 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, researchers soon turned to learning extraction rules from hand-labeled documents. 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) [1] for this purpose. Speech recognition and computational biochemistry are well-known applications of HMMs.

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

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 is to identify unlabeled observations that would be most useful when labeled by the user. Such algorithms are known for classification, clustering, and regression; we present the first algorithm for active learning of hidden Markov models.

2 Using HMMs for IE

The IE task is to attach a semantic *tag* X_i to some of the document tokens O_t. Observations can also be left untagged (special tag $none$). An extraction algorithm f maps an observation sequence $O_1, \ldots, O_T$ to a single sequence of tags (multi-valued assignments would have to be handled by using several IE models, one per label).

There are several natural ways to define the error criterion that an extraction algorithm has to minimize and that quantifies the algorithm's performance; which is appropriate depends on the application. For some applications, costs may arise for each false tag assigned to a token. The *per-token error* is the probability of an extraction algorithm assigning a false tag to a token in the document.

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

Hidden Markov models (see, [2] for an introduction) are a very robust statistical method for structural 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 a token. The tags X_i correspond to the n target states $S_1, \ldots, S_n$ of the HMM. Background tokens without tag 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*).

How can the IE task be addressed with HMMs? Let us first consider what we would do if we already knew an HMM which can be assumed to have generated the observations. Given an observation sequence $O_1, \ldots, O_T$, which tag sequence should we return? If we want to minimize the

*In *Proceedings of the IEEE Internationl Conference on Data Mining.* 2001.

per-token error, then we have to find, for each token, the state in which this token has most likely been emitted (index i with highest $\gamma_t(i)$). If this state is one of the target states, then the corresponding tag has to be assigned to the tokens; the token has to remain untagged otherwise.

When a desired balance between precision and recall is to be achieved, then a token has to be tagged if the probability of the single target state at time t given the observation sequence exceeds a given threshold θ. By adjusting θ, precision can be traded against recall.

Let us now briefly sketch how HMM parameters can be learned from data. In the standard model used in speech recognition, several HMMs are learned (*e.g.,* one for each word to be recognized) and each observation sequence is labeled only with the HMM that it is an example for. In our situation, it is important to achieve a correct assignment of the tokens to target states. We assume that a set of documents is available and the user then labels some of the *tokens* (not whole documents) with the appropriate tag. For each token O_t there is a set of candidate states σ_t; this set may contain one target state (the state corresponding to the label when the user has labeled the token), it can contain all background states (when the user has marked the token as not possessing one of the tags) or it may contain all states (token is unlabeled).

The algorithm which we use to adapt the HMM parameters to the partially labeled documents is an instantiation of the EM algorithm. 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. In the 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 M-step we update the emission and transition probabilities.

In order to execute the E-step of the algorithm, we have to determine $P(q_t = S_i|O_1, \ldots, O_T, \sigma_1, \ldots, \sigma_T)$ – *i.e.,* the probability of being in state S_i at time t given the observation sequence $O_1, \ldots, O_T$ with labels that render states $\sigma_1, \ldots, \sigma_T$ possible. Note that labels can have non-local effects on this probability. Our principle result is a recursive algorithm which determines this probability exactly. The backward-forward-backward algorithm requires three passes over the observation sequence; in the first pass, the probability $\tau_t(i) = P(\sigma_{t+1}, \ldots, \sigma_T|q_t = S_i)$ is computed, in the second and third passes, α and β (which now depend on τ) are calculated. See [3] for details.

We have implemented this algorithm into a web mining tool. The system can be applied to a variety of tasks, such as extracting key information from emails, or spidering business information from the web.

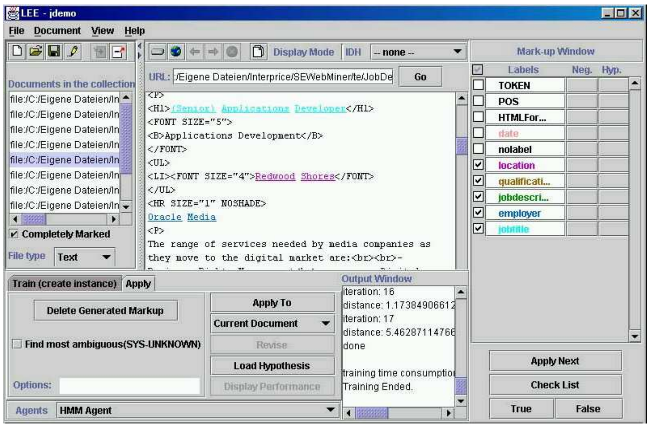

Figure 1. GUI of the SemanticEdge Web Mining tool.

3 Active Revision of Hidden Markov Models

Unlabeled documents can be obtained very easily for most information extraction problems; only labeling tokens in a document imposes effort. Our active learning algorithm selects the most "difficult" unlabeled tokens and asks the user to label them.

We can see the margin $M(q_t|O, \lambda) = \max_i\{P(q_t = S_i|O, \lambda)\} - \max_{j \neq i}\{P(q_t = S_j|O, \lambda)\}$ between the probability of the most likely and that of the second most likely state as the confidence of the state given O. Intuitively, the margin can be seen as quantifying how "difficult" (low margin) or "easy" (large margin) an example is. Our active HMM learning algorithm first learns an initial model λ_1 from a set of partially labeled documents. It then determines the margins of all tokens and starts asking the user to label difficult tokens which have a particularly low margin. The Baum-Welch algorithm is then restarted.

Our experiments [3] show that 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.

References

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

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

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