Hidden Markov Models for Information Extraction

Nancy R. Zhang

June, 2001

Abstract

As compared to many other techniques used in natural language processing, hidden markov models (HMMs) are an extremely flexible tool and has been successfully applied to a wide variety of stochastic modeling tasks. This paper uses a machine learning approach to examine the effectiveness of HMMs on extracting information of varying levels of structure. A stochastic optimization procedure is used to find the optimal structure for a given task, and a modified version of the Baum Welch algorithm is used for parameter estimation.

I. Introduction

The purpose of this project is to investigate the relationship between structure and performance of HMMs applied to information extraction problems. It is intuitive that different state configurations are appropriate for different types of extraction problems. What would be the effect of using the same structural template to train HMMs for different extraction tasks of varying levels of complexity?

A simple structural template will be used for HMM structure learning by the stochastic optimization algorithm described in Freitag & McCallum (1999). Due to the simplicity of the structural template (which will be explained in detail in II.ii), I hypothesize that the search space it defines is sufficient for good performance on simple extraction tasks (extracting fields that are relatively consistent in symbol content and phrase structure), but will not generalize well to harder tasks.

A more important objective of this project is to gain real-life experience in the use of HMMs in information extraction, and to explore the fascinating, less-understood domain of automated structural learning.

II. Methods

II.i Data

The corpus used for this project is created form the L.A. Weekly Restaurants data set taken from the RISE archive of Ion Muslea's group (http://www.isi.edu/~muslea/RISE/). It consists of lists of restaurant information and review in HTML format. The text is semi-structured. The restaurants are organized in HTML list format, with each entry headed by the restaurant name, location, and phone number and followed by the review. The corpus contains listings of restaurants of a variety of different cuisines, which will be useful for the less-structured information retrieval tasks. Each *datapoint* from the corpus consists of the HTML entry for a specific restaurant. Thus, the task in this project is to extract an informational record (e.g. name, phone, cuisine) from each datapoint.

For the purpose of HMM training, the datapoint will be labelled to reflect the target and non-target fields. Accompanying each data symbol sequence will be a binary sequence L_i, with $L_i = 1$ if the ith symbol in the sequence belongs to the target field, and 0 otherwise.

II.ii HMM Structure Template and Learning Algorithm

The structure learning algorithm used is based on that described in Freitag & McCallum. The states in the HMM will be denoted by $S = \{s_1, \ldots, s_n\}$, and the emission symbols by $V = \{ w_1, \ldots, w_K\}$. $O = \{ o_1,\ldots, o_m\}$ represents a the symbol sequence of a single datapoint. The probability of transition to s_j state while in state s_i will be denoted by $P(s_j| s_{1i})$, and that of emitting symbol w_k in state s_i by $P(w_k| s_i)$.

The HMM models used in this study will have four different kinds of states: Background, Prefix, Suffix, and Target. Target states emit tokens for the target field. Prefix and Suffix states emit tokens that appear respectively before and after the target field. All other tokens are emitted in the background state. Each state will contain a label denoting its type.

The structure learning algorithm follows a fixed template in its search and finds the optimal structure from all of the structures reachable given the template. This reduces the search space significantly, and also adds human guidance to the process. Designing the template for the search is no trivial matter, since it determines which structures are and are not considered by the algorithm. The Freitag and McCallum study allows for any number of background states, each having outgoing transition only to the first prefix state and incoming transition only from the last suffix state. This may be useful in the case where the target field is distributed in several regions of the text and must be extracted in fragments, since then the background symbols between the fragments may follow a different distributions. However, in the information extraction tasks that we are considering in this project, each field appears complete and unfragmented and at only one location in each observation sequence. Thus, there is not as much incentive to differentiate between one background state against another, which would significantly increase the complexity of the model. Thus, only one background state is allowed in the template, which would have outgoing transitions to itself and the prefix states, and incoming transitions from the prefix states.

To allow variable prefix length, transition is allowed from the background state to *any* prefix state, and from *any* suffix state to the background state. This is another deviation from the Freitag & McCallum study, which allows transitions only between the background state and the first prefix and final suffix state. This increase in model complexity will prove beneficial, especially for extracting less-structured data. For example, consider the task of extracting the type of cuisine from the restaurant review. The following sequences appear in the corpus:

… light but *traditional* **Mexican** *faire* …
… *intriguing mix of* **Japanese**, **Indian**, and **Italian** *cuisines* …
… *authentic faire from* **Szechuan China** …

If the prefix (suffix) were fixed length, then the further away the prefix (suffix) state from the target state, the more "contaminated" its emission distribution would be by background emissions, and the less useful it would be for capturing target-relevant symbols. Variable length prefix (suffix) allow much more specific information at each prefix state.

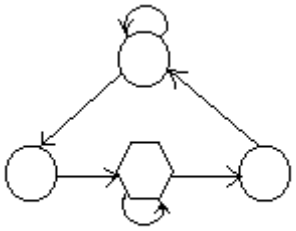


Figure 1

The target states are fully interconnected to allow intricate modeling of the target fields. Other than other target states, each target state can have transition only from the last prefix state, and to the first suffix state. Only one prefix and suffix state sequence is allowed in the model. This simplification from the Freitag & McCallum model compensates for the complexity gain induced by allowing transitions between the background state to any prefix/suffix state. Given this template, the simplest structure would have the configuration shown in figure 1. The allowable steps in the structure space defined by this template would be:

- Lengthen the prefix: add a single state to the beginning of the prefix. The new state will have incoming transition from the background state and outgoing transition to the previously first prefix state.
- Add target state: add a new target state to the collection of interconnected target states. The new state will also have incoming transition from the last prefix state and outgoing transition to the first suffix state, as well as be fully connected to all other target states.
- Lengthen the suffix: add a single state to the end of the suffix. The new state will have incoming transition from the previously last suffix state and outgoing transition to the background state.

Each of the above steps will be referred to as an "operation" in state space. Notice that there is no "backtracking" operations defined. Once a state has been added to the structure, it can not be removed. Thus, each operation always changes a current model to a more complicated model. This poses the problem of local maxima. With no state deletions allowed, it is easier for the search to get stuck in a local maxima. For example, consider the scenario where the best structure has three prefix states and one suffix state, and the current model is the simple model in figure 1. Also, let the model with two prefix states and one suffix state be "worse" than the model with two suffix states and one prefix state. Then, if the model searches in a greedy way, it would take the "lengthen suffix" step and add a suffix state. Once the suffix is added, it can not be deleted in the subsequent operations, and thus the solution found would be suboptimal.

To mitigate the problem of local maximum, at each step the algorithm keeps track of not only the locally "best" operation, but the k best operations. Thus, at each step the algorithm has k "current" models, upon which it applies the operations specified above, obtaining 3k candidate models. It then chooses k of these, and so on. This results in a more thorough search at a cost of k-fold increase in search time.

Following is the algorithm for structure learning. *CurrSet* is the set of current models being analyzed. *Keepers* is the set of all models discovered during the search that we will analyze in the final round to choose the best overall model. *Ops* is the set of operations defined above, and *C* is the number of cross-validation sets used.

Procedure LearnStructure(DataSet)
ValidSet = 1/C of DataSet
TrainSet = (C-1)/C of DataSet
CurrSet = { simplest model }
While all k models in CurrSet have fewer than 25 states

Candidates ← {M| M = op(m), m∈CurrSet, op∈Op}
For M ∈Candidates
Score(M) ← average of 3 runs trained on TrainSet and scored on ValidSet
CurrSet ← k best scorers in Candidates
Keepers ← best scorer in Candidates
For M ∈ Keepers
Score(M) ← average score from 3-fold cross-validation on Labeled Set.
Return M with highest score.

II.iii Parameter Estimation

Parameter Estimation involves optimization of the transition and emission probabilities of a model of fixed configuration. The standard Baum Welch algorithm is used, with slight modifications to reflect the type-constraints of the states. The Baum Welch algorithm is simply the classic Expectation Maximization algorithm applied to HMM learning. The modification to the Baum Welch algorithm needed for this project is the same as that used in Freitag & McCallum, namely:

If TYPE(s) != L_{t+1} then $\alpha_{t+1} = 0$;
Else α_{t+1} is updated as according to the original forward-backward algorithm.

If TYPE(s) != L_t then $\beta_t = 0$;
Else β_t is updated as according to the original forward-backward algorithm.

The transition probabilities are low-degree multinomials for which maximum likelihood estimates suffice. The emission probabilities, however, must be smoothed due to the sparcity of the data relative to the entire English vocabulary. For this study, Lidstone's Law is used to smooth the emission probabilities.

III. Experimental Results

This project attempted to extract the following fields from the restaurants corpus:

(1) Restaurant name
(2) Telephone number
(3) Hours
(4) Cuisine

Fields (1) and (2) are relatively easy. The restaurant name is always somewhere in the beginning of the entry, in bold, large font. The telephone number always follows sortly after, and has a fixed numerical format. The *Hours* field is not so easy, since there is significant variety in the corpus in how the retaurant hours are specified. However, the *Hours* field is also not too hard in that the symbols it emits are limited, and that it always appears somewhere at the end of the entry. The *Cuisine* field is comparatively difficult. There is no rules as to how a restaurant review specify the type of cuisine that the restaurant serves, and sometimes it is not mentioned at all, at which case this field is blank. The relative performance of HMMs in these different extraction tasks will provide valuable insight on the power of HMMs to scale to tasks of increasing levels of difficulty.

For each of the above fields, three-fold cross-validation is used to measure the generalization ability of the HMM trained. The training set is used both to train the structure and estimate its parameters. Given the test set of a sequence of restaurant entries, the HMM must extract each of the symbols that belong to the field under question. Score is tabulated in terms of symbols, and thus if there are *K* symbols belonging to the field for a given entry, the HMM can score between 0 and *K* points on that entry. Performance is measured using the F measure with $\alpha = 0.5$, weighting recall equally against precision.

Table 1 shows the average of the three-fold cross-validatoin scores for each of of the above fields. As expected, the algorithm performs extremely well on the *name* and *telephone number* fields. However, performance both in terms of precision and recall drops drastically for the *hours* and *cuisine* fields. Inspection of performance on training data showed that the models can be trained to have 100% precision and recall on training data, but don't generalize very well to unseen data. This is especially true for the *cuisine* field, where each review presents a new phrase structure for expressing the type of cuisine of the restaurant.

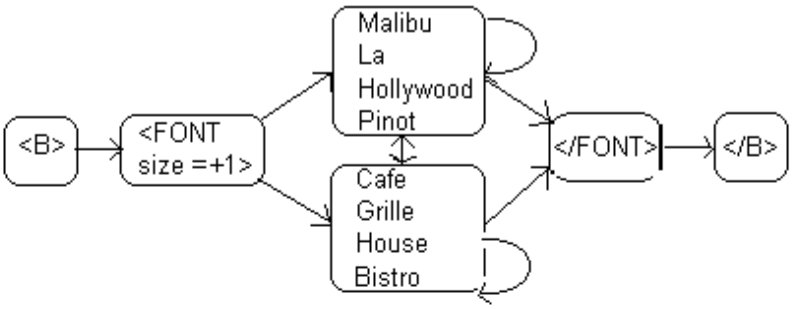
	Precision	Recall	F-Score
Name	0.993	0.975	0.984
Phone number	0.896	0.933	0.914
Hours	0.750	0.826	0.787
Cuisine	0.628	0.547	0.585

Table 1

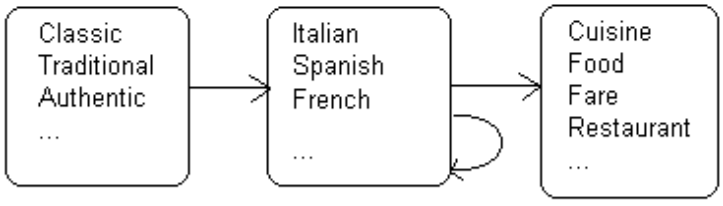
One interesting and unintuitive observation is that more complicated extraction tasks don't always correlate with more complicated HMM structure when trained using a fixed template. For example, the "best" HMM structure for extracting the *cuisine* field found by the structure learning algorithm is the simple structure shown in Figure 1. Since structures are scored for there ability to generalize on unseen data as well as for their accuracy on training data, simple, structures that aim to capture the most important symbols and neglect technical nuances may perform better on fields such as *cuisine* where the unseen data set may differ greatly from the training data. During parameter training, it was also observed that the HMM learned is very specific to the training data set, in that it models exactly what is given to it. Although this guarantees that it does not forego any important detail, it also means that any noise in the training data is also very well incorporated into the model. This suggests that good data pre-processing algorithms and smoothing functions are very important.

Figure 2 shows a simplified representation of some of the "optimal" structures found using the algorithm. Only transitions and emissions with probability greater than 0.5 are shown. Examining the structures, we see that the prefix and suffix states for the "name" field are almost deterministic, in that they predominantly emit only one symble. Thus, despite the fact that the actual target sequence may be irregular, this extraction task is very easy – simply look for a symbol sequence that is bordered by the given prefix and suffix symbols. Due to its simplicity, this task does not make full use of the power of HMMs, that of stochasticity. Examining the HMM for the cuisine field, we see that the opposite is true. The prefix and suffix states are "overcrowded" high order multinomials (only a few of the many symbols emitted in these states are shown). However, the fact that the learning algorithm can not find a better, more complex structure suggests that the template used does not suit this task. For this project, a very simple template is assumed in which only one prefix and suffix state sequence is allowed. A template such as that used in Freitag & McCallum, where parallel sequences are allowed, ought to perform better in extracting the *cuisine* field. However, despite the inadequacy of the simple template

structure, the algorithm has a 0.585 F-score on this task. This is because most of the symbols emitted in the cuisine target field are words such as "Italian", "French", and "Chinese", which are highly identifiable from the background symbols. Thus, the HMM for extracting the name field has highly variable target states and relatively deterministic prefix/suffix states, while that for extracting the cuisine field is just the opposite; it has relatively deterministic target states and highly variable prefix/states. The template structure we proposed in this project assumes sequential prefix and suffix states and fully-interconnected target states, clearly favoring the former task over the latter. This shows that the performance of HMMs on information extraction tasks is highly reliant on the wellness of fit of the state structure to the task-at-hand.



(a) name field



(b) cuisine field

Figure 2

To improve the performance of the models, I experimented with different ways of pre-processing the data. One important consideration is which symbols from the raw data set should be "kept" as an observation. For example, symbols such as "a", "to", and "of" simply contribute extra noise for centain extraction fields such as *name*, but may be important for other fields such as *hour* and *cuisine*. HTML tags are useless for identifying cuisine, but, as shown in figure 2, they are crucial for identifying the *name* field. Thus, any general pre-processor would produce mixed overall benefits.

Also important to the learning process is the parameter *k* in the structure learning algorithm. As stated, *k* is the number of "best" models in the *Candidates* set that are kept in each iteration of the algorithm. Larger *k* allows the exploration of a larger portion of the parameter space. Although the incentive for using larger *k* is logical, in this project it did little to improve performance. This is because the template which defines the search space is already very limited, and thus a broader search does not significantly increase the number of states examined.

IV. Conclusion & Further Work

This was a very challenging yet fun project, through which I learned about the technical, real-life implementation issues in natural language processing that does not surface on the theoretical level. One of the most important conclusions from this project is the high importance of task-directed structure design in HMMs applied to feature extraction. Another insight gained relatively late into the project is that HMMs, although flexible, may not be the best tool for every kind of feature extraction task. For example, the task of extracting the *cuisine* from the restaurant review may be very simple using a vector space clustering approach or an algorithm that simply scans through the review in search of key-words such as "Italian", "French", or "sushi."

Choosing the appropriate data domain and extraction task is critical to the effectiveness of HMMs.

The topic of structural learning of HMMs applied to information extraction holds much potential for further research. Technical issues such as implementing the Baum-Welch algorithm, the structure selection algorithm, and data processing functions comprised the bulk of this project. It would be interesting to experiment with more complicated structure-learning templates, as well as try other smoothing techniques such as the "*Shrinkage*" algorithm proposed in Freitag & McCallum (1999) for adjusting the emission probability distributions. Model training is an extremely slow process that may take up to a day for the more complicated model configurations. With more time and computing power, it might prove insightful to conduct a Monte Carlo simulation of the model likelihood function on the parameter space and examine how it changes relative to model and extraction task complexity. Given more understanding of the characteristics of the likelihood function on the parameter space, we can better evaluate our confidence of the "maximum" likelihood structure returned by the Baum-Welch algorithm.

V. Related Work

Most previous work in the field of information retrieval using HMMs involved hand-built HMMs, such as those described in Bickel et. al. (1998) and Leek (1997). Bickel applied HMMs with machine-learned parameters to the task of finding names and other non-recursive entities in text (the name-entity recognition problem). The program which implemented their hand-coded HMM, Nymble, achieved a high F-score of 90-95. The HMMs implemented by Leek is also intricately designed, for the task of extracting (gene name, chromosome location) pairs from scientific papers in the medical domain.

In fact, HMMs have been applied successfully in many fields related in nature to information extraction. In medicine, HMMs are a crucial tool for extracting "important" DNA fragments from genome data bases (Shatkay et. al. 2000). In phonetics, Markov processes have been used to model the mapping between trajectory segments in acoustic space to phonetic syllables (Saul & Rahim, 1999). The success of HMMs depend on the fact that their graphical representation facilitates human-supervised model design, and yet the existence of EM parameter estimation algorithms allow data-dependent learning.

Much more relevant to this project is the problem of information retrieval using HMMs whose structure is determined by some machine learning algorithm. The primary background of this project is found in the work of Freitag & McCallum (1999) in HMM learning through stochastic optimization . Seymore et. al. (1999) presented a very different approach to HMM structure learning: Start from the most complicated structure and use "merging" techniques to explore the structure space. Such state merging approach is also used by Stolcke and Omohundro (1994).

VI. References

- Bikel D.S., Miller S., Schwartz R. and Weischedel R., *NYMBLE: A High-Performance Learning Name Finder*, In Proceedings of the Fifth Conference on Applied Natural Language Processing, 1997.
- Freitag, D., & McCallum, A., *Information extraction with HMM structures learned by stochastic optimization*. Proceedings of the Eighteenth Conference on Artificial Intelligence (AAAI-2000).
- Freitag D. and McCallum A.L., *Information extraction using hmms and shrinkage*. In Papers from the AAAI-99 Workshop on Machine Learning for Information Extraction, pp 31-36. Jul. 1999. AAAI Techinical Report WS-99-11.
- Leek T.R., *Information extraction using hidden Markov models*. Master's thesis, UC San Diego, 1997.

- Saul L. and Rahim M., *Markov processes on curves for automatic speech recognition*. In Kearns M.S., Solla S. A., and Cohn D.A. (Eds.), Advances in Neural Information Processing Systems, Vol. 11 Cambridge, MA. MIT Press.
- Seymore K., McCallum A., Rosenfeld R., *Learning hidden Markov model structure for information extraction*. In Papers from the AAAI-99 Workshop on Machine Learning from Information Extraction, pp. 37-42. Jul. 1999. AAAI Technical Report WS-99-11.
- Shatkay H., Edwards S., Wilbur W. J. and Boguski M., *Genes, Themes and Microarrays*, Submitted to the Int. Conf. on Intelligent Systems in Molecular Biology, 2000.
- Stolcke A. and Omohundro S.M., *Best-first model merging for hidden Markov induction*. Technical Report TR-94-003, International Computer Science Institute, Berkeley, California, Jan. 1994.