Metadata Extraction from Bibliographies Using Bigram HMM

Ping Yin, Ming Zhang, ZhiHong Deng, and DongQing Yang

School of Electronics Engineering and Computer Science,
Peking University, Beijing, China
{yinping_, mzhang, zhdeng, ydq}@db.pku.edu.cn

Abstract. In recent years, we have seen huge volumes of research papers available on the World Wide Web. Metadata provides a good approach for organizing and retrieving these useful resources. Accordingly, automatic extraction of metadata from these papers and their bibliographies is meaningful and has been widely studied. In this paper, we utilize a bigram HMM (Hidden Markov Model) for automatic extraction of metadata (i.e. title, author, date, journal, pages, etc.) from bibliographies with various styles. Different from the traditional HMM, which only uses word frequency, this model also considers both words' bigram sequential relation and position information in text fields. We have evaluated the model on a real corpus downloaded from Web and compared it with other methods. Experiments show that the bigram HMM yields the best result and seem to be the most promising candidate for metadata extraction of bibliographies.

1 Introduction

Authors and publishers are beginning to make scientific publications available on the World Wide Web in increasing number. In order to search and exploit these disorganized digital documents, there is a growing need to organize them efficiently. Organizing articles by their metadata is a good way and becomes more and more popular. Accordingly, automatic extraction of metadata from vast number of papers and papers' bibliographies has been widely studied in recent years. We are interested in improving the metadata extraction from papers' bibliographies, which is, segmenting a bibliography into individual fields such as author, title, publisher, date and so on.

The field extraction from bibliographies is non-trivial because of the high variance in the structure of the current record-level search. Previous approaches have typically used rule-based system to do this. Citeseer [2] uses a heuristic method which first parses those fields that have relatively uniform syntax, position, and composition. In addition, it uses syntactic relationships between fields and dictionaries of author names and journal titles to help identify fields. There is also another rule-based bibliographic metadata extractor called DECITER (decoding citations) [3]. There are some problems in such systems which rely on hand-written rules. Firstly, rules have to be modified if an entry with a new style is added to the domain. Secondly, they only work for the regions they are developed and can't extend to other domains. A lot of manual work has to be performed in rewriting these rules while shifting domains. In this paper, we adopt a bigram HMM to automatically extract bibliographic metadata with a seed set of example labeled bibliography entries.

The remainder of the paper is organized as follows. Section 2 describes Hidden Markov Models as background. Section 3 describes the key steps for extracting metadata from bibliographies via a bigram HMM. Section 4 experimentally evaluates the bigram HMM on a corpus. Section 5 discussed some related works. Section 6 summarizes the paper.

2 Hidden Markov Models

A Hidden Markov model (HMM) is a finite state automation comprising with stochastic state transitions and symbol emissions. The automation models a probabilistic generative processes whereby a sequence of symbols is produced by starting at a designated start state, transitioning to a new state, emitting a symbol selected by that state, transitioning again, emitting another symbol, and so on, until a designated final state is reached. Associated with each of a set of states, $S = \{S_1, ..., S_n\}$, are a probability distribution over the symbols in the emission vocabulary $V = \{w_1, ... w_m\}$, and a probability distribution over its set of outgoing transitions. [4, 5]

In this model, a symbol sequence can be generated through some state path with a probability which can be computed as the product of all transition and emission probabilities along the path. Given an output sequence, we can also recover the most probable state transitions that could have generated it.

HMMs, while relatively new to the structure extraction task, have been used with much success for speech and hand-writing recognition tasks and for natural language tasks like parts-of-speech tagging. In spite of the general principles being known, applying it to information extraction requires new enhancements to this model.

Next section we will see how to implement the metadata extraction using a modified HMM, bigram HMM.

3 Bibliographic Metadata Extraction with Bigram HMM

A bibliographic entry can be viewed as a sequence of fields (e.g. author, title, publisher, date, pages, etc.). Given an HMM, each state of which is marked with a label that is the name of some field., metadata extraction from bibliographies is performed by determining the sequence of states that was most likely to have generated the entire word sequence of the bibliography entry, and then putting each word to the corresponding field according to the state sequence.

To perform extracting we therefore require an algorithm for finding the most likely state sequence given a HMM model M and a sequence of symbols. Although a naïve approach for finding the most likely sequence would take time exponential in the sequence length, a dynamic programming solution called the viterbi algorithm [1, 4] solves the problem in just $O(TN^2)$ time.

To perform extracting we also need to build an HMM, including the structure and the parameters. Other work such as OOV problem, parameter smoothing and so on has to be dealt with as well to finish the extracting perfectly.

3.1 The Viterbi Algorithm

Given an output sequence $O = O_1 O_2 ... O_T$ of length T and an HMM having N states, we want to find out the most probable state sequence from the start state to the end state which generates O. [1, 4]

Let S_0 and S_{N+1} denote the special start and end states which don't emit symbols.

Let $\delta_t(j)$ denotes the highest probability along a single path, at time t, which accounts for the first t observations and ends in state S_j. Therefore $\delta_t(j)$ can be written as

$$\delta_t(j) = \max_{q_1 q_2 \cdots q_{t-1}} P(q_1 q_2 ... q_t = S_j, O_1 O_2 ... O_t \mid \lambda) \ . \tag{1}$$

We begin at the start state S_0. Thus, initially,

$$\delta_0(0) = 1,\ \delta_0(k) = 0,\ k \neq 0 \ . \tag{2}$$

By induction we have

$$\delta_t(j) = \max_{1 \le i \le N} \left[\delta_{t-1}(i) a_{ij} \right] b_j(O_t),\ 1 \le t \le T, 1 \le j \le N \ , \tag{3}$$

where a_{ij} is the transition probability from state S_i to state S_j, $b_j(O_t)$ is the emission probability of emitting O_t at state S_j. The maximum is taken over all states of the HMM.

Finally, that is at time *T+1*, the state sequence will end at the end state S_{N+1}. So we have

$$\delta_{T+1}(N+1) = \max_{1 \le i \le N} \delta_T(i) a_{i(N+1)}, \qquad \delta_{T+1}(j) = 0,\ 1 \le j \le N \ . \tag{4}$$

The most probable path can be gotten by storing the argmax at each step. This formulation can be easily implemented as a dynamic programming algorithm running in $O(TN^2)$ time.

3.2 Learning Structure

In order to build an HMM for information extraction, first of all, we must decide how many states the model should contain and what transitions between states should be allowed. A reasonable initial model is to use one state per field, and to allow transitions from any state to any other state. However, this model may not be optimal in all cases. When a specific hidden sequence structure is expected in the extraction domain, we may do better by building a model with multiple states per field, with only a few transitions out of each state. This can be done by learning the structure automatically from the labeled training data consisting of labeled word sequences. [6]

Firstly an HMM is constructed which produces exactly the input word sequences. The start state has as many outgoing transitions as there are word sequences and each word sequence is represented by a unique path with one state per word. All paths end at the final state with probability 1. The probability of entering these paths from the start state is uniformly distributed. Within each path there is a unique transition arc whose probability is 1. The emission probabilities are 1 from each state to produce the corresponding word. This model is called as maximum likelihood model.[6]

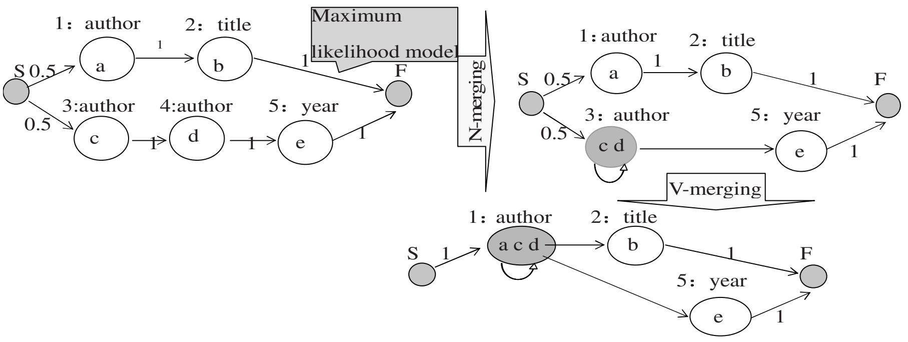

Fig. 1. Learning structure using labeled bibliography entries "<author>a</author><title> b</title>" and "<author>c d</author><year>e</year>"

Then "Neighbor-merging" and "V-merging" are used to merge some states that have the same label to generalize the maximally specific model. "Neighbor-merging" combines all states that share a link and have the same field label. "V-merging" merges any two states that have the same label and share transitions from or to a common state.

All of these can be illustrated in Fig. 1.

3.3 Learning Parameters

Once the structure of the HMM is fixed, we need to learn its transition and emission probabilities, which can be calculated using the Maximum Likelihood approach on all training sequences.

The probability of making a transition from state q to state q' is the ratio of the number of transitions made from state q to q' in the training data to the total number of transitions made from q. The probability of emitting symbol σ at state q is the ratio of the number of times σ is emitted in q to the total number of symbols emitted in the state. This can be written as

$$P(q \to q') = \frac{c(q \to q')}{\sum_{s \in Q} c(q \to s)} \qquad P(q \uparrow \sigma) = \frac{c(q \uparrow \sigma)}{\sum_{\rho \in \Sigma} c(q \uparrow \rho)} \tag{5}$$

where $q \uparrow \sigma$ denotes state q emits word σ and $c(x)$ denotes the number of event x occurring in the training data. [6]

In this model, it is unreasonable that two words with the same frequency in the same state have equal importance because it ignores much helpful information. Firstly, it ignores any sequential relationship amongst words in the same filed. For example, phrases like "Technical Report" will be outputs of the same state. This state will accept "Technical Report" with the same probability as "Report Technical". In fact, it is very unusual that "Report Technical" appears. Secondly, it ignores the words' position information within a filed. For example, "pp." is more important than any other word (i.e. "w") that occurs in the same frequency with "pp.", because "pp." always appears in the beginning of the pages filed, while "w" always appears inside the pages field. In practice, we will pay more attention to the words that always occur in the beginning of a field. However, this model can't distinguish this, which will treat "pp" and "w" as with the same importance.

We overcome these drawbacks by using a bigram HMM in the next section.

3.4 Bigram HMM

The Bigram uses a modified model for computing the emission probability, while keeping the structure of HMM unchanged. In the new model, the probability of emitting symbol σ at state q composes of beginning emission probability and inner emission probability. The former is the probability that q emits σ as the first word, and the latter is the probability that q emits σ as the inner word (not the first word). We will later see how to use the emission probability in the modified viterbi algorithm and understand that by this we can capture the words' position information in the state. The inner emission probability is computed using a bigram model, which can capture the words' bigram sequence relationship within the same filed.

We can write the new emission probability model as

$$P(\sigma \mid q) = \begin{cases} P(q \underline{\uparrow} \sigma), & \sigma \text{ appears in the beginning of q} \\ P(q \overline{\uparrow} \sigma) = P(\sigma \mid \sigma_{-1}, q), & \sigma \text{ appears in the inner of q} \end{cases} \tag{6}$$

where $q \underline{\uparrow} \sigma$ denotes state q emits word σ as the beginning word, $q \overline{\uparrow} \sigma$ denotes q emits σ as the inner word and σ_{-1} denotes the word before σ.

We can also use the ratio from the training data to compute $P(q \nwarrow \sigma)$ and $P(\sigma \mid \sigma_{-1}, q)$.

$$P(q \underline{\uparrow} \sigma) = \frac{c(q \underline{\uparrow} \sigma)}{\sum_{\rho \in \Sigma} c(q \underline{\uparrow} \rho)} \qquad P(\sigma \mid \sigma_{-1}, q) = \frac{c(q \uparrow \sigma_{-1}\sigma)}{c(q \uparrow \sigma_{-1})} \tag{7}$$

3.5 Viterbi for Bigram HMM

We make a little modification to the viterbi algorithm for the bigram HMM, where we will see how to use the beginning emission probability and the inner emission probability presented in last section.

$$\delta_t(j) = \max_{1 \le i \le N}\left[\delta_{t-1}(i)a_{ij}\right]b_j(O_t)\ , \qquad 1 \le t \le T, 1 \le j \le N$$
$$\text{wher e}\ ,\ b_j(O_t) = \begin{cases} P(S_\text{j} \underline{\uparrow} O_t)\ , & S_j \ne S_{j\text{-}1} \\ P(O_t \mid O_{t\text{-}1}, S_j)\ , & S_j = S_{j\text{-}1} \end{cases} \tag{8}$$

Only the formula in the induction has been modified, where the beginning emission probability is used if the current state S_j is not equal to the last state $S_{j\text{-}1}$ which shows the current word O_t is the first word of state S_j, or else, the inner emission probability is used.

3.6 Smoothing

When the training data is insufficient, maximum likelihood estimation of emission probabilities will lead to poor estimates, with many words inappropriately having zero probability. So we need to smooth emission probabilities to prevent zero-probability estimates and improve estimation overall. There are many methods for smoothing. Both *Laplace smoothing* and *absolute discounting* calculate the word distribution in a state using only the training data in state q itself. In contrast, the third technique called *shrinkage* can leverage the word distributions in several related states in order to improve parameter estimation.[5]

An idea similar to *shrinkage*, a method which we call as *back off-shrinkage* [7] can be used here for the bigram HMM. Because of insufficient training data, bigram HMM may not see some bigrams or words, in which case the model backs off to a less-powerful, less-descriptive model. So we define a level of back-off models as below, from bigram model to unigram model, then to global model, and finally to uniform model.

Backing off of $P(q \underline{\uparrow} \sigma)$: $P(q \underline{\uparrow} \sigma) \to P(q \uparrow \sigma) \to P_{global}(\sigma) \to 1/m$

Backing off of $P(\sigma \mid \sigma_{-1}, q)$: $P(\sigma \mid \sigma_{-1}, q) \to P(q \uparrow \sigma) \to P_{global}(\sigma) \to 1/m$

where m is the size of vocabulary, $1/m$ is the emission probability of the uniform model, and $P_{global}(\sigma)$ is the emission probability in the global model, that is the probability of σ occurring in the training data.

We then combine the estimates with a weighted average, for example,

$$\begin{aligned} &\bar{P}(q \underline{\uparrow} \sigma) = \lambda_1 P(q \underline{\uparrow} \sigma) + \lambda_2 P(q \uparrow \sigma) + \lambda_3 P_{global}(\sigma) + \lambda_4 / m, \\ &\text{where } \lambda_1, \lambda_2, \lambda_3, \lambda_4 \ge 0\ ,\ \text{and}\ \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1 \end{aligned} \quad . \tag{9}$$

The weights are adjusted in practice according to the actual importance of models. In the bibliographic metadata extraction, the words in the beginning of fields are very important, so λ_1 was assigned a large value in the experiment. Because the global emission probability $P_{global}(\sigma)$ is always non-zero, the uniform emission probability is meaningless. Therefore, we set λ_4 to 0.

3.7 OOV Problem

During testing we may encounter words that have not been seen during training. How we estimate the emission probabilities of these unknown words. This problem is known as OOV (out of vocabulary) problem. This paper uses a method which we call as *minimum frequency method* [6]. Let f denotes the minimum frequency. The vocabulary is constituted by the words whose frequency in the training data is not less than f , while other words whose frequency is less than f are mapped to the unknown word "<UNK>". Any word in the testing data that is out of vocabulary is also mapped to "<UNK>", so its emission probability is the emission probability of "<UNK>" which can be estimated via the training data.

4 Experimental Evaluation

4.1 Preliminaries

713 bibliography entries were stochastically extracted from 250 papers using the paper metadata extractor we have implemented before, which can extract title, author, abstract, keywords, the list of bibliographies and so on from PDF formatting papers. These entries were then hand-labeled as follows to construct our data set.

"<author>Andrew W. Appel. </author><title>A semantic model of types. </title><journal>In Twenty-Seventh ACM Symposium, </journal><pages>pages 243-253, </pages><location>Boston, </location><date>January 2000. </date>"

We use 4-level cross validation, splitting the dataset into four parts averagely, one part as testing set in turn, other three parts merged as training set, performing four times of experiments. The training set is used to train the HMM, and the testing set is used to evaluate the effect of extraction. The final result is the average of the four experiments' results.

We use both precision (P) and recall (R) borrowed from information-retrieval community to evaluate our result where

$$P=\frac{\text{number of tokens corrected tagged using HMM}}{\text{number of tokens tagged using HMM}} \text{ and } R=\frac{\text{number of tokens corrected tagged using HMM}}{\text{number of tokens tagged by expert}}$$

F1, the harmonic mean of P and R, i.e. $2/(1/P+1/R)$, is used to balance P and R.

4.2 Results

We find the smoothing method *back off-shrinkage* makes better precision than other methods, so we choose it as the smoothing method in our experiment.

In Figure 2 we compare the precision, recall and F1 of three different models, bigram HMM presented in this paper, traditional HMM (unigram HMM) and DECITER mentioned in section 1 as a rule-based metadata extractor. We can make the following observations from these results.

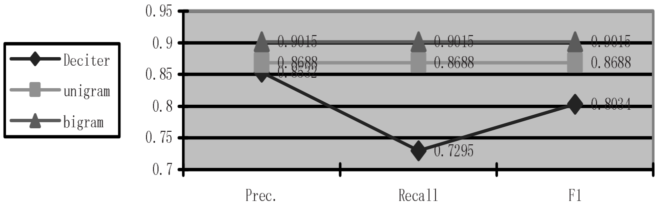

Fig. 2. Comparison of different models

Table 1. Results in Individual fields

Field	Tokens present	Prec.	Recall
title	2957	0.8959	0.9212
author	3144	0.9770	0.9587
date	616	0.9239	0.9447
pages	570	0.9552	0.9630
volume	162	0.8405	0.8997
issue	135	0.8860	0.8995
journal	2221	0.9052	0.8080
url	218	0.8770	0.9553
publisher	78	0.6436	0.8330
location	345	0.6689	0.7827
other	174	0.4288	0.6180
total	10620	0.9015	0.9015

1. The overall precision and overall recall are equal when using both traditional and bigram HMM because all tokens are tagged, while not equal when using DECITER because DECITER leaves many tokens untagged by not assigning them to any of the fields, which also causes the low recall 0.7295.
2. Bigram HMM makes an improvement in precision more than three percentages than traditional HMM, while DECITER gets the lowest precision and recall. The peak precision is more than 90%, which is satisfactory.

Finally, details of precision and recall in every field when using bigram HMM are shown in table 1. The precision and recall of most fields are acceptable except the three fields publisher, location and other which accepts tokens that don't belong to any other filed. Fortunately, these fields happen to be less important and occur infrequently in training and testing data. The scarcity of data prevents them from getting trained properly.

5 Related Work

There is much work related to ours.

[8] also uses HMM to extract metadata from bibliographies as this paper. However, unlike this paper, in [8] the structure of HMM is hand-crafted according to some bibliographic style, so the result is very bad when using this model to extract bibliographies with other styles. In contrast, this paper can automatically learn the HMM structure using the labeled training data in which there are bibliography entries with as much styles as possible, and can achieve high precision in spite of the style of the testing data.

[7] presents a Nymble system to perform "named entity" extraction as defined by MUC-6 using a two-level hierarchical HMM in which the nested model is a full-connected model which can also overcome the shortcoming presented in section 3.3. All different fields to be extracted are modeled in a single HMM composed of some name-class states, and each name-class state is composed of m (the size of vocabulary) word-states each of which generates the corresponding word with probability 1 and can connect to any other word-state.

[1] presents a DATAMOLD system also using a two-level hierarchical HMM to segment text into structured records, an application of which is also the bibliographic metadata extraction. Unlike [7], the outer model and inner model can both be learned from the training data. The inner HMM can capture the finer structure of the corresponding filed, and it can also capture the length information of the field by a parallel path structure.

In contrast to [7] and [1], this paper keeps the structure of HMM unchanged, capturing part structures of fields by modifying the emission probability model and accordingly the viterbi algorithm.

6 Conclusions

This paper has dedicated to the problem of extracting metadata from bibliographic entries using a bigram HMM which uses a modified emission probability model to exploit additional cues from several sources, including words' sequential relation and words' position information within a filed. The structure and parameters of HMM is automatically learned from the labeled training data, alone with which is the *back-off shrinkage* smoothing. Experiments yield precision greater than 90%, considerably better than traditional HMM and a rule-based algorithm.

References

1. V. R. Borkar, K. Deshmukh, and S. Sarawagi. Automatic Segmentation of Text into Structured Records, Proc. ACM-SIGMOD Int'l Conf. Management of Data (SIGMOD 2001), ACM Press, New York, 2001, pp. 175-186.
2. S. Lawrence, C. Giles, and K. Bollacker, "Digital libraries and autonomous citation indexing," IEEE Computer, vol. 32, no. 6, pp. 67-71, 1999.

3. harvester.jar, http://www.cs.cornell.edu/cdlrg/Reference%20Linking/software/RefLink.tar.gz
4. L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE, 1989, 77(2), pp. 257-285.
5. Freitag D., McCallurn A. Information extraction with HMMs and shrinkage, Workshop Notes of AAAI-99 Conference on Machine Learning for Information Extraction, 1999, pp. 31-36.
6. Seymore K., McCallum A., Rosenreid R. Learning hidden Markov model structure for information extraction. AAAI-99 Workshop on Machine Learning for Information Extraction, 1999, pp. 37-42.
7. Bikel D.M., Miller S., Schwartz R. and Weischedel R. Nymble: a high performance learning namefinder. In Proceeding of the fifth Conference on Applied Language Processing, 1999, pp. 194-201.
8. J. Connan and C.W. Omlin. Bibliography Extraction with Hidden Markov Models, Technical Report US-CS-TR-00-6, 24 February 2000, Department of Computer Science, University of Stellenbosch.
9. Leek T. Information Extraction Using Hidden Markov Models, Masters Thesis, Department of Computer Science & Engineering, University of California, San Diego, 1997.
10. Freitag, D., McCallum, A. Information extraction with HMM structures learned by stochastic optimization. Proceedings of the Eighteenth Conference on Artificial Intelligence (AAAI-2000), 2000.
11. Stolcke, A. and Omohundro, S.M. Hidden Markov Model Induction by Bayesian Model Merging. Advances in Neural Information Processing Systems, 1992, Volume 5, S. J. Hanson, J. D. Cowan and C. L. Giles, editors, Morgan Kaufman, pp. 11-18, 1992.
12. Andreas Stolcke and Stephen M. Omohundro. Best-first model merging for hidden Markov model induction. Technical Report TR-94-003, Computer Science Division, University of California at Berkeley and International Computer Science Institute, 1994.
13. A. McCallum, D. Freitag, and F. Pereira. Maximum entropy Markov models for information extraction and segmentation, in Proc. 17th International Conf. on Machine Learning, 2000, pp. 591-598.
14. Probabilistic Logic Learning Seminar. Hidden Markov Models for Information Extraction.
15. Soderland S. Learning Information Extraction Rules for Semi-Structured and Free Text, Machine Learning: Special Issue on Natural Language Learning, 1999, 34, pp. 233-272.