Address Extraction Using Hidden Markov Models

Kazem Taghva, Jeffrey Coombs, Ray Pereda, Thomas Nartker
Information Science Research Institute
University of Nevada, Las Vegas
Las Vegas, NV 89154-4021, USA

ABSTRACT

This paper presents the implementation and evaluation of a Hidden Markov Model to extract addresses from OCR text. Although Hidden Markov Models discover addresses with high precision and recall, this type of Information Extraction task seems to be affected negatively by the presence of OCR text.

Keywords: Information Extraction, Hidden Markov Models, OCR

1. INTRODUCTION

In this paper we describe the development and effectiveness of a Hidden Markov Model (HMM) based extraction program to identify potentially private addresses. We found that HMM's provide high recall and precision for this task. However, we also discovered that optical character recognition (OCR) errors degraded the performance of the program. This last result is somewhat surprising given that document retrieval has been shown, on average, not to be so affected.[1,2] We also discovered that for our task, the smoothing technique called *shrinkage* did not improve our results.

We will first describe the problem our HMM is designed to solve in section 2. We discuss previous attempts to extract information such as addresses using HMM's in section 3. Presented in section 4 is a standard definition of HMM's and the details of our implementation of an address-finding HMM. In section 5 we describe how the HMM was trained. We next discuss some problems resulting from errors in the OCR processing of training documents in section 6. In section 7 we describe the construction and outcomes of experiments measuring the effectiveness of several variations on our address finding HMM for various types of documents. Finally, in section 8, our conclusions and prospects for future experimentation are offered.

2. BACKGROUND

According to the Privacy Act of 1974, the United States Government cannot make public private information about American citizens.[3] However, to comply with the Freedom of Information Act, the Federal Government must make many of its documents and records available to the public. These documents, therefore, must be reviewed to determine if they contain private information. Since the number of documents to be reviewed is very large, and human review is costly and time-consuming, it is clearly advantageous to seek automatic methods for identifying likely private information in documents.

For certain types of private information, such as social security numbers, a simple regular expression matching technique proved effective for the majority of cases. However, the prospects of such a solution for identifying addresses seemed unlikely as we reviewed the data and noted the wide variation in address formats. In addition, many documents had been OCR-ed and some noise had been introduced into the target addresses. Since Hidden Markov Models have been applied to problems similar to ours with some success, we decided to explore the use of an HMM to identify personal addresses.[4,5]

Our task is to identify a set of documents such that the likelihood of them containing private information is extremely low. In other words we want to locate the "easy" examples of non-private documents so that they can be released in accordance with the Freedom of Information Act.

Documents that we place in the "private" category will very likely still contain documents without private information. However, these more difficult cases will be subject to a manual review by human experts. Our automatic process aims to reduce the number of documents requiring human review but does no eliminate the need for such a review completely.

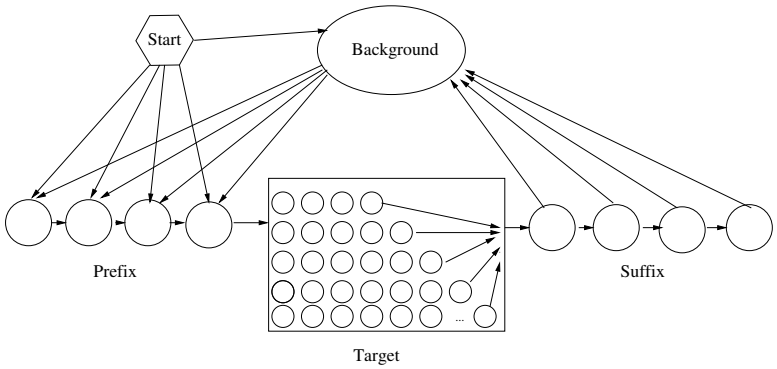

Figure 1. Hidden Markov Model for Addresses

Thus, as we will see in section 8, the worst mistake for us is a false negative: a document containing private information in which our HMM finds none. We can accept, within reason, a higher rate of false positives if doing so decreases false negatives. We can accept loses in precision to boost recall.

3. PREVIOUS WORK

Our approach belongs to a family of tasks called *information extraction* (IE). We understand information extraction to be the automatic identification of text sequences from a document which fulfill some specific information need. Some[6] require that the information found be used for filling in the fields of a database, but such a definition seems too narrow since the information discovered could have many uses. For example, in our case we use the information extracted to aid in the categorization of a document.

One may roughly distinguish two approaches to IE using HMM's. The first assumes that a document has been, and thus can be, partitioned in some way. In the work of Leek[5] and Bikel,[4] for example, documents had been separated into sentences. Borkar[7] assumes that addresses have already been identified prior to applying his HMM which identifies address segments, such as "City" and "Street." Unlike Borkar's task of finding address segments within pre-identified addresses, our problem was to discover likely addresses within whole documents.

The second approach models the entire document, and the HMM is designed to extract desired features without prior partitioning. Since some of our documents were OCR-ed texts, we did not want to rely on any Natural Language Processing, even to the extent of only identifying sentences. Hence we decided to follow Freitag's and McCallum's approach of using an HMM to model the entire document.[6]

4. HIDDEN MARKOV MODELS

A Hidden Markov Model is a finite state automaton with probabilistic transitions and symbol emissions. An HMM consists of

- A set of states, S = $\{s_1, \cdots s_n\}$
- An emission vocabulary V = $\{w_1 \cdots w_n\}$
- Each state is associated with a probability distribution over emission symbols where the probability that a state s emits symbol w is given by $P(w|s)$.
- Each state is further associated with a probability distribution over the set of possible outgoing transitions. The probability of moving from state s_i to s_j is given by $P(s_i, s_j)$.
- Finally, a subset of the states are considered start states, and to each of these is associated an "initial" probability that the state will be a start state.

symbol	*examples*
`comma`	final comma (,)
`colon`	final colon (:)
`ziplike`	89783, 89123-2319
`phonelike`	(555) 555-5555, 555-555-5555, 555-5555
`purenumber`	5, 5555
`containsnumber`	n5k
`mailterm`	mail,P.O.,address,apt
`roadname`	street, road, avenue, ave
`statename`	nevada, nv
`cityname`	las vegas, oak ridge
`pname`	joe, smith
`startcap`	Yucca, Mountain
`default`	yucca, tree, mountain

Figure 2. HMM Symbols

Following the model of Freitag and MacCallum,[6] our HMM has four types of states. *Background* states represent symbols that are not part of an address nor part of the "context" of one. We defined the "context" to include the four terms in the document prior and the four terms subsequent to the address. The prior four terms correspond to what we call the four *prefix* states of an address, and the following four terms in the context are four *suffix* states. *Target* states represent the parts of the address. Figure 1 provides an illustration of the structure of our HMM. (More specifically, figure 1 illustrates the structure of our "clean" HMM as discussed in section 7.)

Figure 2 displays how we defined the emission vocabulary for the HMM. As in Bikel[4] vocabulary identification had an ordering which is reflected in the list in our figure. This means that a term is first scanned to determine if is ends in a comma. If so, the comma is removed and the comma itself is represented in the HMM with the distinct symbol `comma`. Next, the remainder of the term is checked to determine if it is a five or nine digit number with an optional hyphen after the fifth digit. In such a case the term is represented in the HMM with the symbol `ziplike`. (The symbol `colon` is only added if a term ends in a colon prior to the removal of a final comma.) If it is not such a number, then it is checked if it is `phonelike`, and so on.

Terms were placed in this taxonomy in two ways. Either a straightforward regular expression match was used, as in the case of `ziplike` and `phonelike`, or a term was matched against a database of known objects. For example, a very short list of city names most relevant to the documents was generated. This list took into consideration that most of the documents concerned government employees and contractors from government offices and businesses dealing with nuclear power.

After a document is converted to symbols by this taxonimizing process, addresses are discovered by applying the well-known Viterbi algorithm to the resulting string of symbols. The Viterbi algorithm uses dynamic programming techniques to determine the most likely state sequence for the document.[8] Of interest to us are those subsequences consisting of target states since they represent the most likely addresses.

5. TRAINING

Training for the HMM was based on tagged documents which were marked by human experts with the Information Science Research Institute's (ISRI) META-Marker tool.[9] The topology of transitions as illustrated in figure 1 was predetermined except for the lengths of the target sequences, which were determined by the marked data. Figure 1 is somewhat simplified for aesthetic purposes. Transitions from suffix states to any of the prefix states was allowed, and a final target state could transition to a prefix state if addresses appeared close together in training data.

Transition probabilities were estimated by Maximum Likelihood:

$$P(s_i, s_j) = \frac{\text{Number of transitions from } s_i \text{ to } s_j}{\text{Total number of transitions out of } s_i} \tag{1}$$

Initial probabilities representing the beginning of the document were set by hand. The initial probability was divided uniformly over the background state, all prefix states, and the first target state.

The emission probability table is estimated with Maximum Likelihood supplemented by *smoothing*. Smoothing is required because Maximum Likelihood estimation will sometimes assign a zero probability to unseen emission-state combinations.

Prior to smoothing, emission probabilities are estimated by:

$$P(w|s)_{ml} = \frac{\text{Number of times symbol } w \text{ is emitted at state } s}{\text{Total number of symbols emitted by state } s} \tag{2}$$

We used *absolute discounting* to smooth emission probabilities. Absolute discounting consists of subtracting a small amount of probability p from all symbols assigned a non-zero probability at a state s. Probability p is then distributed equally over symbols given zero probability by the Maximum Likelihood estimate.

If v is the number of symbols assigned non-zero probability at a state s and N is the total number of symbols, emission probabilities are calculated by

$$P(w|s) = \begin{cases} P(w|s)_{ml} - p & \text{if } P(w|s)_{ml} > 0 \\ \frac{vp}{N-v} & \text{otherwise} \end{cases} \tag{3}$$

There is no known solution for determining the optimal value for p.[6] We followed Borkar[7] in using $\frac{1}{T_s+v}$ where T_s is the total number of symbols emitted by a state s, i.e., the denominator of $P(w|s)_{ml}$.

In addition to absolute discounting, Freitag and McCallum[6] propose applying a more elaborate smoothing technique called *shrinkage*. According to this approach, various alternative distributions are defined for the emission probability $P(w|s)$ for each state s. The *default* distribution is absolute discounting. Further, each probability can "shrink" to its *parent* distribution. For example, the target parent is estimated by dividing the number of times a symbol is emitted by *any* target state by the total number of symbols emitted by any target state. The non-target *ancestor* shrinks to the common ancestor state of all background, prefix, and suffix states. The *context grandparent* shrinks prefix and suffix states to a state common to all prefixes and suffixes. Finally each distribution can shrink to the *uniform* distribution.

The following table gives the types of shrinkages possible for each state.

background	default	background parent	non-target ancestor	uniform	
target	default	target parent	uniform		
suffix	default	suffix parent	context grandparent	non-target ancestor	uniform
prefix	default	prefix parent	context grandparent	non-target ancestor	uniform

For each symbol-state combination, the final emission probability $\hat{P}(w|s)$ is determined by a weighted sum of the estimates from the table above. For example, a given suffix state will have a default probability, a suffix parent, a context grandparent, a non-target ancestor, and the uniform distribution probabilities associated with it, and each of these will be assigned a weight $\{\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5\}$. We thus define

$$\hat{P}(w|s) = \sum_{i=1}^{k} \lambda^i P(w|s^i) \tag{4}$$

In the case of a suffix state, for example, $\{P(w|s^1), P(w|s^2), P(w|s^3), P(w|s^4)P(w|s^5)\}$ would represent its default, suffix parent, context grandparent, non-target ancestor, and uniform probabilities. The value k is the number of alternative probabilities for a given type of state. For a suffix state, k=5.

Optimal values for the λ weights are discovered using Expectation Maximization. Following Freitag[6] we initialize the λ's to $\frac{1}{k}$. Holding out each symbol in turn in a hold-out set $\mathcal{H}_j$, we perform the following two steps until the values of all λ's no longer change:

E-step. Determine how much each emission state predicts the symbols from a held-out set $\mathcal{H}_j$ for each state s:

$$\beta^i = \sum_{w \in \mathcal{H}_j} \frac{\lambda^i P(w|s^i)}{\sum_m \lambda^m P(w|s^m)} \tag{5}$$

M-step. Derive new λ's by normalizing the β's:

$$\lambda^i = \frac{\beta^i}{\sum_m \beta^m} \tag{6}$$

Finally, the output of our HMM consists of sequences of strings which are likely addresses. In fact, the HMM will sometimes discover non-addresses, and the HMM has no concept of a "private" address. For these reasons, filters had to be introduced to remove obvious non-addresses and known "public" addresses.

6. OCR RELATED DIFFICULTIES

Several difficulties occurred in OCR documents which affected the training for the HMM.

- While the META-marker tool allowed experts to mark zones on the document image which were derived from OCR information, the *ordering* of the text in the OCR output did not always correspond to the ordering inferred by the human expert viewing the image. Figure 3 shows a redacted private address in the context of handwritten text. Figure 4 reveals how the OCRed text was tagged. Prefix-tagged terms were separated from target-tagged terms by background-tagged terms and the suffix-tagged terms were taken out of order. In fact, the term tagged as "suffix3" was placed in the midst of background-tagged terms, and a single address was tagged as two distinct addresses. Obviously, there is only a tenuous correspondence between the tagged address and the "ideal" topology of figure 1. ("X" is used in figure 4 to redact private information.)
- In some documents, addresses viewable in the document images were either partially or completely lost in the OCR process. Also, useful context information would occasionally be lost as well. Figure 5 and its corresponding OCR in figure 6 provide an example of lost information.
- In some documents *handwritten* addresses appeared which were not recognized by the OCR. Several documents were simply removed from our training and testing sets because these problems recognizing handwriting appear to us to require a different approach entirely.

7. EXPERIMENTS

The first two problems noted in section 6 raise the question whether the tagging should be "cleaned up" to reflect our preconceived ideas about how the HMM should be structured, or whether it should be left "as is" to reflect the reality of the data.

We decided to compare the performance of an HMM trained on tagged documents which contained "incorrect" tags with an HMM trained only on documents where the tags fit our assumed topology illustrated in figure 1.

Human experts identified 251 documents and emails in a large collection of documents from various government agencies which contain addresses subject to the privacy act. Addresses in 187 of these documents were tagged with the METAmarker tool. Documents with handwritten addresses were removed from the 187 and 89 were randomly selected for training what we call the *default* HMM. Some of these 89 documents contain "incorrect" tagging due to the OCR difficulties described in section 6.

A second training set was derived from the first by removing all examples which contained "incorrect" tagging. The HMM based on this training set was called *clean*.

A test set of 614 documents and emails was constructed from (1) a random sample from the 251 documents containing private addresses, with hand-written examples removed, (2) a random sample of emails from a large (more than 20,000) collection of government emails, and (3) a random sample from 2,000 government documents.

Three experiments were performed. In the first, the default HMM was run on the test set. In the second, the clean HMM was run against the same data. Finally, the emission table for the clean HMM was optimized using shrinkage.

The standard performance measures of precision, recall, and F1 were calculated for each experiment. Let TP be the number of true positives, that is, the number of documents which both experts and the HMM agreed contain private addresses. Let FN be the number of false negatives, i.e., the number of documents which experts said contain privacy, but the HMM marked as not having privacy. We then define *recall* as

$$recall = \frac{TP}{TP + FN} \tag{7}$$

Letting FP signify the number of false positives, i.e., those documents which the HMM marked as containing privacy but which experts decided does not, *precision* is defined as

$$precision = \frac{TP}{TP + FP} \tag{8}$$

The harmonic mean of precision and recall is called the $F1$ measure, defined as:

$$F1 = \frac{2}{1/precision + 1/recall} \tag{9}$$

The results of our experiments are presented below.

experiment	TP	FP	FN	TN	precision	recall	$F1$
default	104	4	9	495	0.963	0.920	0.941
clean	108	16	5	483	0.871	0.956	0.911
shrinkage	108	28	5	471	0.794	0.956	0.867

The value TN represents the number of true negatives, which are documents which both the experts and the HMM agreed did not contain private addresses.

8. CONCLUSION AND FUTURE WORK

Our testing indicates that shrinkage does not significantly improve an HMM's performance for this task. In fact, there was some degradation.

More interesting is the relative "success" of the default HMM. This may suggest that for OCR texts the structure of HMM's should be more flexible than that of the Frietag-MacCallum HMM (figure 1). For example, perhaps there should be an additional "error state" accessible to all states with a low transition probability that would recognize addresses with "out of order" elements such as in the tagged data of figure 4.

For our task, however, the "clean" HMM is preferable to the others. As discussed in section 2, correctly finding documents with private addresses is, to a certain degree, more important than mistakenly marking non-private documents as private.

Although it has been shown that OCR errors do not affect the average effectiveness of information retrieval,[1, 2] some studies have indicated that noisy data can degrade information extraction tasks such as text summarization.[10] Our test results indicate that noisy data will also degrade tasks such as searching a text for segments belonging to some specific category, such as addresses.

There seems to be some "common sense" support for this conclusion. When retrieving OCR documents, a search engine will still have a large proportion of the text available to it due to the high accuracy of OCR

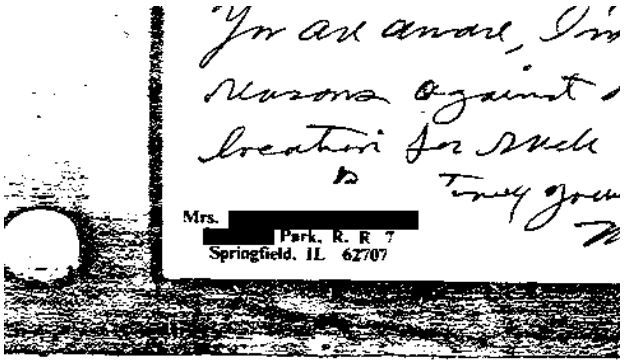

Figure 3. Address in HQZ.19880629.6150

engines. However, information extraction requires that specific, localized patterns be found. If the very objects being sought have been lost to the OCR process, extraction will fail.

We plan to further study the effects of OCR noise on IE tasks by comparing the performance of HMM's on clean and noisy text collections in various IE tasks.

We also hope to further investigate the variation of certain parameters of HMM's. For example, Freitag suggests a version of absolute discounting different from Borkar's,[7] which we used. We would like to determine in later experiments which is superior.

REFERENCES

1. K. Taghva, J. Borsack, and A. Condit, "Effects of OCR errors on ranking and feedback using the vector space model," *Inf. Proc. and Management* **32**(3), pp. 317–327, 1996.
2. K. Taghva, J. Borsack, and A. Condit, "Evaluation of model-based retrieval effectiveness with OCR text," *ACM Transactions on Information Systems* **14**, pp. 64–93, January 1996.
3. "The privacy act of 1974 5 u.s.c. sec. 552a." Url: `http://www.usdoj.gov/foia/privstat.htm`. Viewed June 14, 2004.
4. D. Bikel, S. Miller, and R. Weischedel, "Nymble: a high-performance learning name-finder," in *Proceedings of ANLP-97*, pp. 194–201, 1997.
5. T. Leek, "Information extraction using hidden markov models," Master's thesis, UC San Diego, 1997.
6. D. Freitag and A. McCallum, "Information extraction with HMMs and shrinkage," in *Proceedings AAAI-99 Workshop Machine Learning and Information Extraction*, 1999.
7. V. Borkar, K. Deshmukh, and S. Sarawagi, "Automatic segmentation of text into structured records," in *ACM SIGMOD 2001*, (Santa Barbara, California, USA), 2001.
8. L. Rabiner and B. Juang, *Fundamentals of Speech Recognition*, Prentice Hall, 1993.
9. K. Taghva, M. Cartright, and J. Borsack, "An efficient tool for xml data prepartation." Forthcoming in the Proceedings of Information Systems: Next Generation, 2004.
10. H. Jing, D. Lopresti, and C. Shih, "Summarizing noisy documents," in *Proceedings of SDIUT'03*, pp. 111–119, (Greenbelt, MD), April 2003.

```
<background>Mrs.</background>
<prefix1>X.</prefix1>
<prefix2>XXXX</prefix2>
<prefix3>XXXXX</prefix3>
<prefix4>XXXXXX</prefix4>
<background>’’</background>
<background>^--^^^</background>
<background>^2</background>
<background>-^</background>
<background>r^</background>
<target1>XX</target1>
<target2>XXXX</target2>
<target3>Park.</target3>
<target4>R.</target4>
<target5>R.</target5>
<target6>7</target6>
<background>/vim.</background>
<background>t3</background>
<suffix3>v^’’^</suffix3>
<background>^^</background>
<target1>Sprin</target1>
<target2>eld.</target2>
<target3>IL</target3>
<target4>62707</target4>
<suffix1><-1.17-</suffix1>
<suffix2>^</suffix2>
<suffix4>1</suffix4>
<background>c</background>
```

Figure 4. Tagging of HQZ.19880629.6150

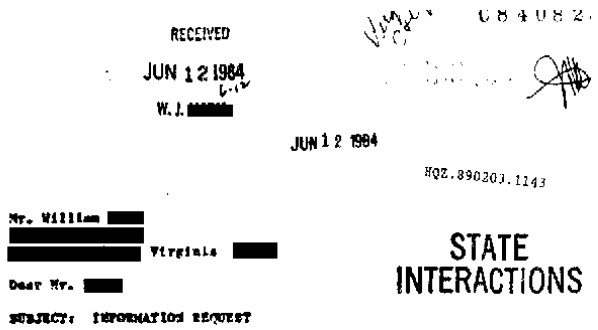

Figure 5. Address in HQZ.19890203.1143

```
Mr. ’teaks RECEIVED JUN 121964 W. 1.
XXXXX TiritinLs 21163 WUICTt REAMT w ^v C’, / / C8 408 25 AOo.
JUN 12 1984 STATE INTERACTIONS This is is
```

Figure 6. Resulting OCR of HQZ.19890203.1143