Maximum Entropy Markov Models for Information Extraction and Segmentation

Andrew McCallum, Dayne Freitag, and Fernando Pereira

17th International Conf. on Machine Learning, 2000

Presentation by Gyozo Gidofalvi
Computer Science and Engineering Department
University of California, San Diego
gyozo@cs.ucsd.edu
May 7, 2002

Outline

- Modeling sequential data with HMMs
- Problems with previous methods: motivation
- Maximum entropy Markov model (MEMM)
- Segmentation of FAQs: experiments and results
- Conclusions

Background

- A large amount of text is available on the Internet
  - We need algorithms to process and analyze this text
- Hidden Markov models (HMMs), a “powerful tool for representing sequential data,” have been successfully applied to:
  - Part-of-speech tagging:

    <PRP>He</PRP> <VB>books</VB> <NNS>tickets</NNS>
  - Text segmentation and event tracking:

    tracking non-rigid motion in video sequences
  - Named entity recognition:

    <ORG>Mips</ORG> Vice President <PRS>John Hime</PRS>
  - Information extraction:

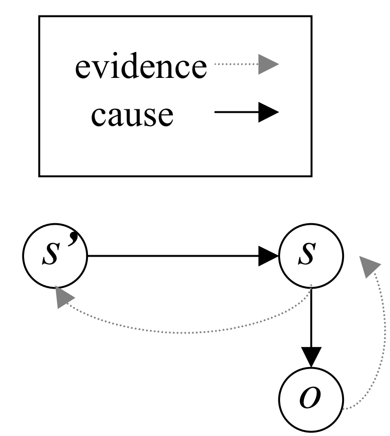
    <TIME>After lunch</TIME> meet <LOC>under the oak tree</LOC>

Brief overview of HMMs

- An HMM is a finite state automaton with stochastic state transitions and observations.

- Formally: An HMM is
  - a finite set of states ***S***
  - a finite set of observations ***O***
  - two conditional probability distributions:
    - for s given s': $P(s|s')$
    - for o given s: $P(o|s)$
  - the initial state distribution $P_0(s)$

Dependency graph



The "three classical problems" of HMMs

- **Evaluation** problem: Given an HMM, determine the probability of a given observation sequence $\overline{o} = \langle o_1, \ldots, o_T \rangle$:

$$P(\overline{o}) = \sum_{\overline{s}} P(\overline{o} \mid \overline{s}) P(\overline{s})$$

- **Decoding** problem: Given a model and an observation sequence, determine the most likely states that led to the observation sequence $\overline{s} = \langle s_1, \ldots, s_T \rangle$:

$$\arg\max_{\overline{s}} P(\overline{o} \mid \overline{s})$$

- **Learning** problem: Suppose we are given the structure of a model (***S***, ***O***) only. Given a set of observation sequences determine the best model parameters.

$$\arg\max_{\theta} P(\overline{o}, \theta) = \sum_{\overline{s}} P(\overline{o} \mid \overline{s}, \theta) P(\overline{s})$$

- Efficient dynamic programming (DP) algorithms that solve these problems are the Forward, Viterbi, and Baum-Welch algorithms respectively.

Assumptions made by HMMs

- **Markov assumption**: the next state depends only on the current state
- **Stationarity assumption**: state transition probabilities are independent of the actual time at which transitions take place
- **Output independence assumption**: the current output (observation) is independent of the previous outputs (observations) given the current state.

Difficulties with HMMs: Motivation

- We need a richer representation of observations:
  - Describe observations with overlapping features
    - When we cannot enumerate all possible observations (e.g. all possible lines of text) we want to represent observations by feature values.
  - Example features in text-related tasks:
    - capitalization
    - word ending
    - part-of-speech
    - formatting
    - position on the page

Example task:

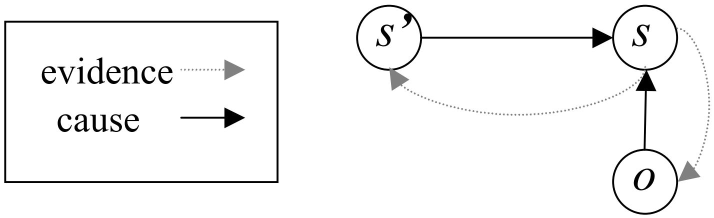
Extract company names

- Model $\underbrace{P(s_T|o_T)}_{\text{Discriminative / Conditional}}$ rather then the joint probability $\underbrace{P(s_T,o_T)}_{\text{Generative}}$

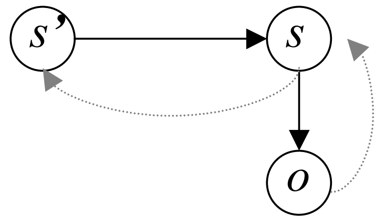
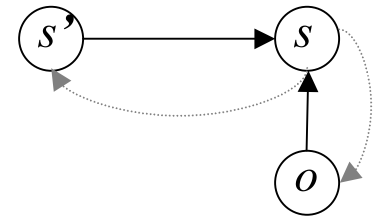
Definition of a MEMM

- Model the probability of reaching a state given an observation and the previous state

Dependency graph



- finite set of states ***S***
- set of possible observations ***O***
- State-observation transition probability for s given s' and the current observation o: $P(s|s',o)$
- initial state distribution: $P_0(s)$

Task	Generative: HMM 	Discriminative / Conditional: MEMM 
Evaluation	Find P(o_T|M)	
Decoding = Prediction	Find s_T s.t. P(o_T| s_T, M) is maximized	Find s_T s.t. P(s_T| o_T, M) is maximized
Learning	Given ***o***, find *M* s.t. P(***o*** | *M*) is maximized (Need EM because S is unknown)	Given ***o*** and ***s***, find *M* s.t. P(***s*** | ***o***, *M*) is maximized (Simpler Max likelihood problem)

DP to solve the "three classical problems"

- $\alpha_t(s)$ is the probability of being in state s at time t given the observation sequence up to time t:

$$\alpha_{t+1}(s) = \sum_{s' \in S} \alpha_t(s') \cdot P(s \mid s', o_{t+1}) \qquad (1)$$

- $\beta_t(s)$ is the probability of starting from state s at time t given the observation sequence after time t:

$$\beta_t(s') = \sum_{s \in S} P(s \mid s', o_t) \beta_{t+1}(s) \qquad (2)$$

Maximum Entropy Markov Models (MEMMs)

- For each s' separately conditional probabilities $P(s|s',o)$ are given by an exponential model
- Each exponential model is trained via maximum entropy

Note: $P(s|s',o)$ can be split into $|\mathbf{S}|$ separately trained transition functions $P_{s'}(s|o) = P(s|s',o)$.

Fitting exponential models by maximum entropy

- Basic idea:
  - The best model of the data satisfies certain constraints and makes the fewest possible assumptions.
  - "fewest possible assumptions" ≡ closest to the uniform distribution (i.e. has highest entropy)

- Allow non-independent observation features
- Constraints are counts for properties of training data:
  - "observation contains the word apple" and is labeled "header"
  - "observation contains a capitalized word" and is labeled "question"
- Properties (called features) can depend on observations and also their state label.
- Formally: A feature f_a is defined by $a = \langle b, r \rangle$, where
  - b is a binary feature of the current observation and
  - r is a state value:

$$f_{\langle b,r \rangle}(o_t, s_t) = \begin{cases} 1 & \text{if } b(o_t) \text{ is true and } s_t = r \\ 0 & otherwise \end{cases} \qquad (3)$$

Constraints on the model

- For all s' the expected value E_a of each feature a in the learned distribution equals its average value F_a in training set:

$$E_a = \frac{1}{m_{s'}} \sum_{k=1}^{m_{s'}} \sum_{s \in S} P(s \mid s', o_k) f_a(o_k, s) = \frac{1}{m_{s'}} \sum_{k=1}^{m_{s'}} f_a(o_k, s_k) = F_a \quad (4)$$

- Theorem: The probability distribution with maximum entropy that satisfies the constraints is (a) unique, (b) the same as the ML solution, and (c) in exponential form. For a fixed *s'*:

$$P(s \mid s', o) = \frac{1}{Z(o, s')} \exp\left(\sum_a \lambda_a f_a(o, s) \right) \quad (5)$$

where λ_a are the parameters to be learned and

$$Z(o, s') = \frac{P(s \mid s', o)}{\sum_{s \in S} P(s \mid s', o)} \quad (6)$$

MEMM training algorithm

1. Split the training data into observation - destination state pairs ⟨*o*,*s*⟩ for each state *s'*.
2. Apply Generalized Iterative Scaling (GIS) for each *s'* using its ⟨***o***,***s***⟩ set to learn the maximum entropy solution for the transition function of *s'*.

This algorithm assumes that the state sequence for each training observation sequence is known.

GIS [Darroch & Ratcliff, 1972]

- Learn the transition function for one origin state *s'* by finding λ_a values that satisfy $E_a = F_a$ (Eq 4).
- Input for one origin state *s'*:
  - training examples with this origin *s'* numbered 1 to *k*
  - for each of these training examples
    - set of features f_a for $a = 1 \ldots n$
      - values for features for each context $\langle o,s \rangle$ must sum to constant C
      - Use correction feature f_x if necessary: $f_x(o,s) = C - \sum_{a=1}^{n} f_a(o,s)$
- Outputs: set of λ_a values for $a = 1 \ldots n$

For a fixed *s'*:

1. Let m_s be the number of training examples where the current state is *s* (and the previous state is *s'*).
2. Calculate the relative frequency of each feature on the training data:

$$F_a = \frac{1}{m_s} \sum_{k=1}^{m_s} f_a(o_k, s_k) \tag{7}$$

3. Initialize λ_a to some arbitrary value, say 1.
4. Use current λ_a values in Eq 5 to estimate P(*s*|*s'*,*o*)
5. Calculate the expectation of each feature "according to the model":

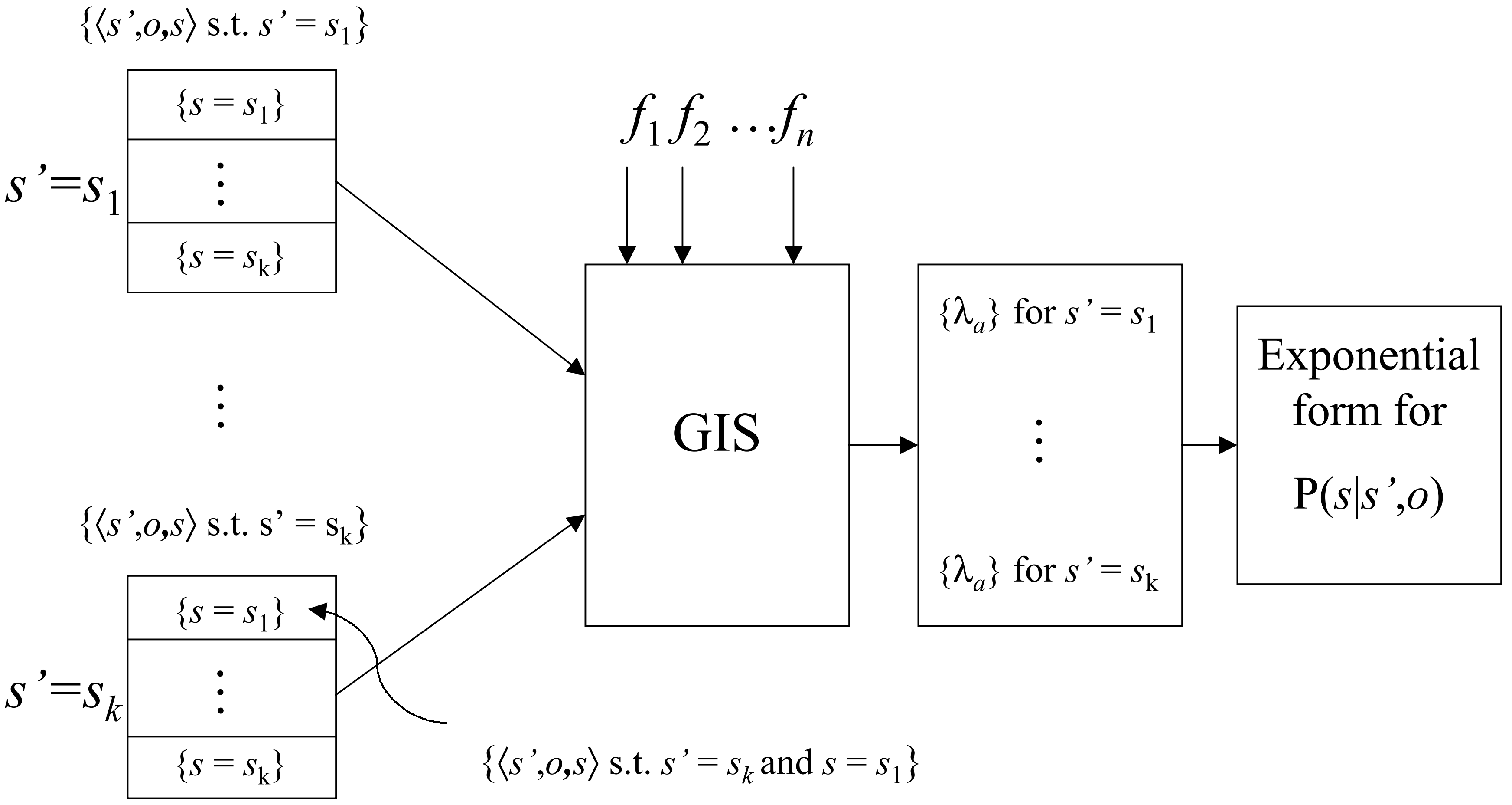
$$E_a = \frac{1}{m_s} \sum_{k=1}^{m_s} \sum_{s \in S} P(s \mid s', o_k) f_a(o_k, s) \tag{8}$$

6. Update each λ_a s.t. to make E_a be closer to the expectation of the training data:

$$\lambda_a := \lambda_a + \frac{1}{C}(\log F_a - \log E_a) \tag{9}$$

7. Repeat from step 4 until convergence.

Review of the MEMM model



Application: segmentation of FAQs

- 38 files belonging to 7 Usenet multi-part FAQs (set of files)
- Basic file structure:

```
header
   text in Usenet header format
   [preamble or table of content]
series of one of more question/answer pairs
tail
   [copyright]
   [acknowledgements]
   [origin of document]
```

- Formatting regularities: indentation, numbered questions, types of paragraph breaks
- Consistent formatting within a single FAQ

- Lines in each file are hand-labeled into 4 categories: *head*, *questions*, *answers*, *tail*

```
<head>X-NNTP-Poster: NewsHound v1.33
<head>
<head>Archive-name: acorn/faq/part2
<head>Frequency: monthly
<head>
<question>2.6) What configuration of serial cable should I use
<answer>
<answer> Here follows a diagram of the necessary connections
<answer>programs to work properly. They are as far as I know t
<answer>agreed upon by commercial comms software developers fo
<answer>
<answer> Pins 1, 4, and 8 must be connected together inside
<answer>is to avoid the well known serial port chip bugs. The
```

Table 2: An excerpt from a labeled FAQ

- Prediction: Given a sequence of lines, a learner must return a sequence of labels.

Boolean features of lines

- The 24 line-based features used in the experiments are:

begins-with-number
begins-with-ordinal
begins-with-punctuation
begins-with-question-word
begins-with-subject
blank
contains-alphanum
contains-bracketed-number
contains-http
contains-non-space
contains-number
contains-pipe
contains-question-mark
contains-question-word
ends-with-question-mark
first-alpha-is-capitalized
indented
indented-1-to-4
indented-5-to-10
more-than-one-third-space
only-punctuation
prev-is-blank
prev-begins-with-ordinal
shorter-than-30

Experiment setup

- "Leave-*n*-minus-1-out" testing: For each file in a group (FAQ), train a learner and test it on the remaining files in the group.
- Scores are averaged over $n(n$-1) results.

Evaluation metrics

- *Segment*: consecutive lines belonging to the same category
- *Co-occurrence agreement probability* (COAP)
  - Empirical probability that the actual and the predicted segmentation agree on the placement of two lines according to some distance distribution *D* between lines.

$$P_D(actual, predicted) = \sum_{i,j} D(i,j) \begin{bmatrix} actual(i) = actual(j) \\ = \\ predicted(i) = predicted(j) \end{bmatrix}$$

  - Measures whether segment boundaries are properly aligned by the learner
- *Segmentation precision* (SP): $\dfrac{\text{\# of correctly identified segments}}{\text{\# of segments predicted}}$

- *Segmentation recall* (SR): $\dfrac{\text{\# of correctly identified segments}}{\text{\# of actual segments}}$

Comparison of learners

- **ME-Stateless**: Maximum entropy classifier
  - documents is an unordered set of lines
  - lines are classified in isolation using the binary features, not using label of previous line
- **TokenHMM**: Fully connected HMM with hidden states for each of the four labels
  - no binary features
  - transitions between states only on line boundaries
- **FeatureHMM**: same as TokenHMM
  - lines are converted to sequences of features
- **MEMM**

Results

Learner	*COAP*	*SegPrec*	*SegRecall*
ME-Stateless	0.520	0.038	0.362
TokenHMM	0.865	0.276	0.140
FeatureHMM	0.941	0.413	0.529
MEMM	0.965	0.867	0.681

References

- McCallum, A., & Freitag, D., & Pereira, F., (2000). Maximum Entropy Markov Models for Information Extraction and Segmentation. Proc. 17th International Conf. on Machine Learning pp. 591-598.
- A Brief MAXENT tutorial:
  http://www-2.cs.cmu.edu/afs/cs/user/aberger/www/html/tutorial/tutorial.html