Hidden Markov Models

Hrant Baloyan, Khachatur Inants

Hidden Markov Models (HMMs) are an extension of Markov Chains.

Markov Chain consists of

1) A sequence of states

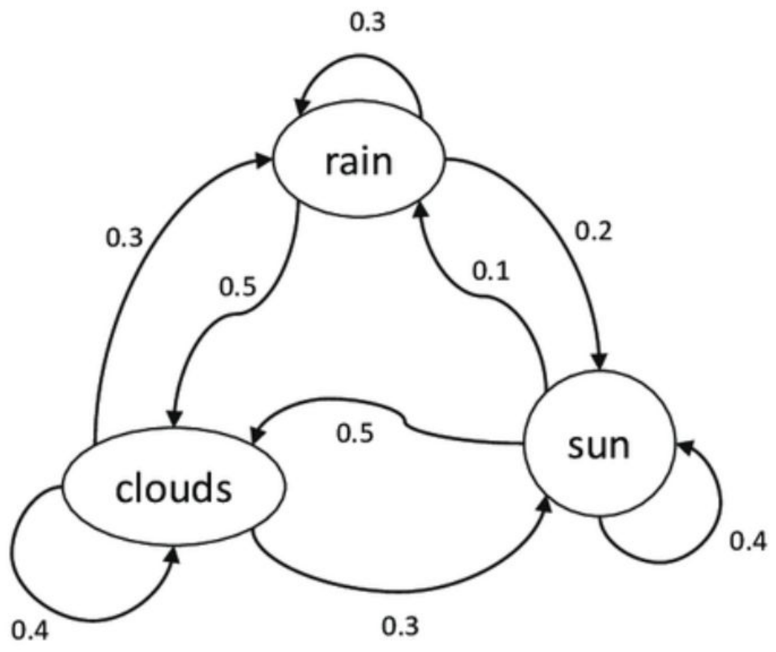
2) A transition matrix

3) An initial probability distribution

Main property

Markov Assumption: $P(q_i = a | q_1 ... q_{i-1}) = P(q_i = a | q_{i-1})$

Example of Markov Chain



$$
\begin{array}{c|ccc}
 & \text{clouds} & \text{rain} & \text{sun} \\
\hline
\text{clouds} & 0.4 & 0.3 & 0.3 \\
\text{rain} & 0.5 & 0.3 & 0.2 \\
\text{sun} & 0.5 & 0.1 & 0.4
\end{array}
$$

But what if we have hidden events?

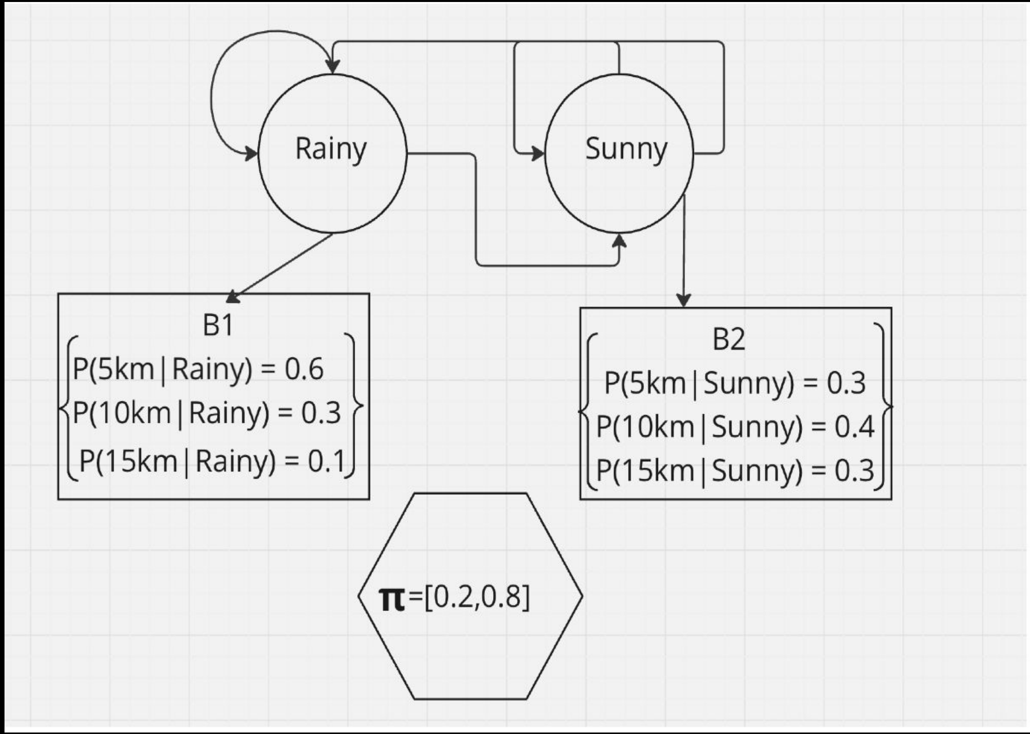
HMM has

1) Sequence of hidden states

2) Transition matrix

3) Probabilities that represent what's the probability of observation o being generated by state q

4) An Initial probability distribution



HMM makes two assumptions

Markov assumption:

$$P(q_i|q_1 \cdots q_{i-1}) = P(q_i|q_{i-1})$$

Independence assumption:

$$P(o_i|q_1 \cdots q_i \cdots q_T, o_1, \cdots o_i, \cdots o_T) = P(o_i|q_i)$$

HMMs can be described in 3 problems.

1) Likelihood - Given HMM $\lambda = (A, B)$ and sequence of observations O, find $P(O|\lambda)$

1) Decoding - Given sequence of observations O andHMM $\lambda = (A, B)$, find best sequence of hidden states

1) Likelihood - Given HMM $\lambda = (A, B)$ and sequence of observations O, find $P(O|\lambda)$

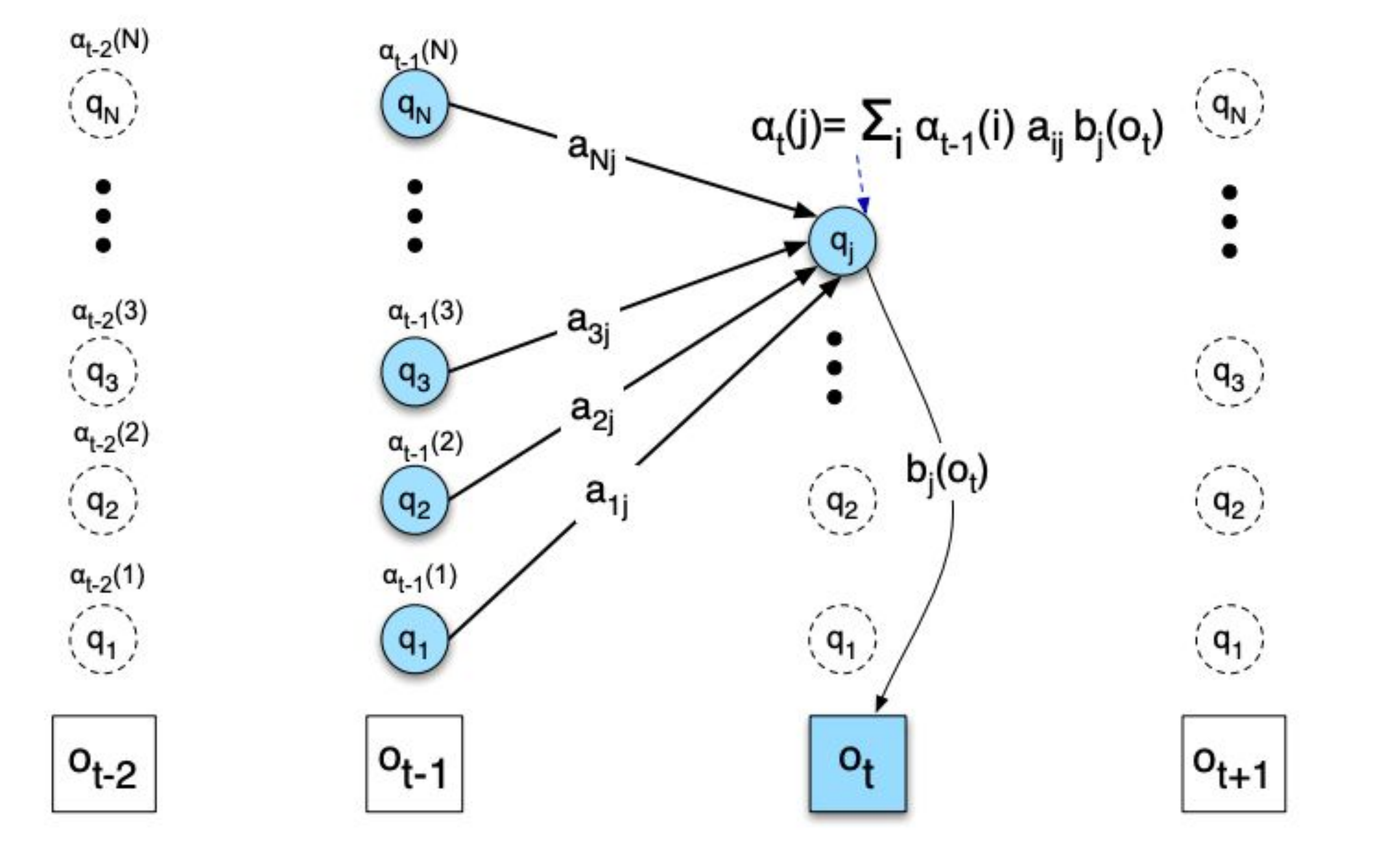
Likelihood Computation : The Forward Algorithm

Problem is to compute the likelihood of a particular observation sequence.

Each cell of the forward algorithm trellis $\alpha_t(j)$ represents the probability of being in state j after seeing the first t observations,

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t)$$

$\alpha_{t-1}(i)$ the previous forward path probability from the previous time step
a_{ij} the transition probability from previous state q_i to current state q_j
$b_j(o_t)$ the state observation likelihood of the observation symbol ot given the current state j



αt-2(N)
qN
αt-1(N)
qN
aNj
αt(j)= Σi αt-1(i) aij bj(ot)
qN
qj
αt-2(3)
q3
αt-1(3)
q3
a3j
q3
a2j
αt-2(2)
q2
αt-1(2)
q2
a1j
q2
bj(ot)
q2
αt-2(1)
q1
αt-1(1)
q1
q1
q1
ot-2
ot-1
ot
ot+1

function FORWARD(*observations* of len T, *state-graph* of len N) **returns** *forward-prob*

create a probability matrix *forward[N,T]*
for each state s **from** 1 **to** N **do** ; initialization step
$forward[s,1] \leftarrow \pi_s \, * \, b_s(o_1)$
for each time step t **from** 2 **to** T **do** ; recursion step
for each state s **from** 1 **to** N **do**

$$forward[s,t] \leftarrow \sum_{s'=1}^{N} forward[s',t-1] \, * \, a_{s',s} \, * \, b_s(o_t)$$

$$forwardprob \leftarrow \sum_{s=1}^{N} forward[s,T]$$; termination step

return *forwardprob*

The Viterbi algorithm

Given a series of observed events, the Viterbi algorithm determines the most likely order of hidden states in an HMM

$v_t(j)$, represents the probability that the HMM is in state j after seeing the first t observations and passing through the most probable state sequence $q_1, \ldots, q_{t-1}$,

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t)$$

$v_{t-1}(i)$ the previous Viterbi path probability from the previous time step

a_{ij} the transition probability from previous state q_i to current state q_j

$b_j(o_t)$ the state observation likelihood of the observation symbol ot given the current state j

$bt_t(j)$, represents which state at a time t-1 maximize the probability of reaching the state j after seeing the observation t.

$$bt_t(j) = \operatorname*{argmax}_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t)$$

for each state s **from** 1 **to** N **do** ; initialization step

$viterbi[s,1] \leftarrow \pi_s * b_s(o_1)$

$backpointer[\mathrm{s},1] \leftarrow 0$

for each time step t **from** 2 **to** T **do** ; recursion step

for each state s **from** 1 **to** N **do**

$$viterbi[\mathrm{s,t}] \leftarrow \max_{s'=1}^{N} \; viterbi[s',t-1] * a_{s',s} * b_s(o_t)$$

$$backpointer[\mathrm{s,t}] \leftarrow \operatorname{argmax}_{s'=1}^{N} \; viterbi[s',t-1] * a_{s',s} * b_s(o_t)$$

$$bestpathprob \leftarrow \max_{s=1}^{N} \; viterbi[s,T]$$; termination step

$$bestpathpointer \leftarrow \operatorname{argmax}_{s=1}^{N} \; viterbi[s,T]$$; termination step

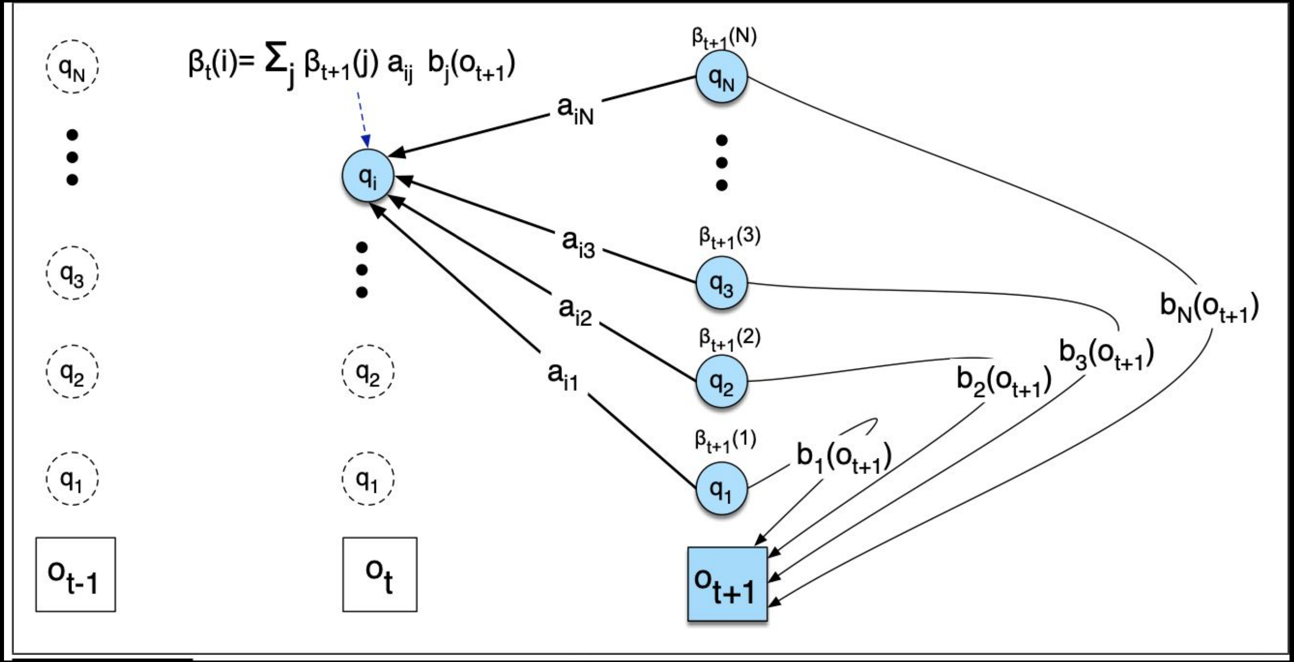
bestpath $\leftarrow$ the path starting at state *bestpathpointer*, that follows backpointer[] to states back in time

return *bestpath*, *bestpathprob*

Learning of HMM: Forward-Backward algorithm

Here we are trying to find parameters of HMM A and B using observations and states

Introducing backward probabilities



Need we try to estimate aij

$$\hat{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}$$

Let's define

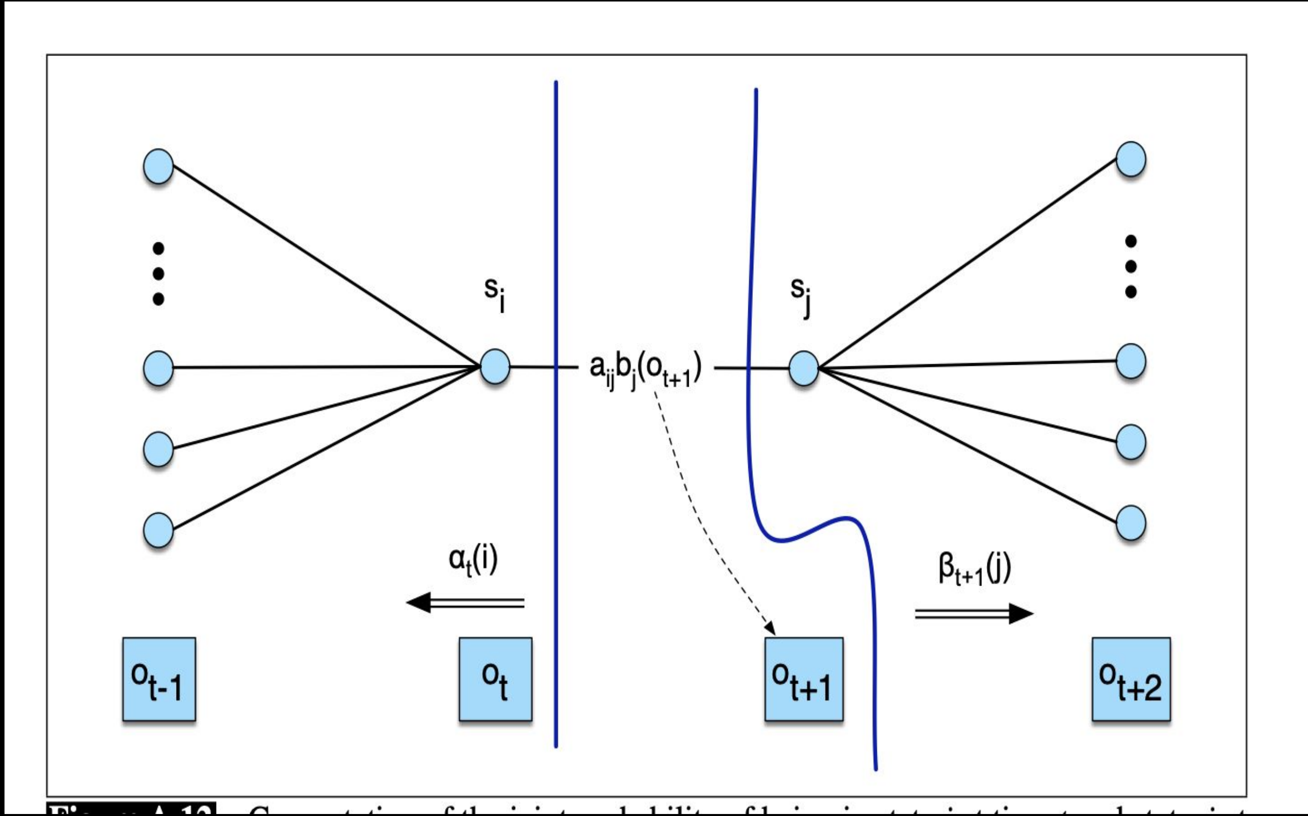
$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j | O, \lambda)$$

First let's compute

$$\text{not-quite-}\xi_t(i,j) = P(q_t = i, q_{t+1} = j, O | \lambda)$$

Continue

$$\text{not-quite-}\xi_t(i,j) = \alpha_t(i)\, a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$$



Using $$P(X|Y,Z) = \frac{P(X,Y|Z)}{P(Y|Z)}$$

We compute ξt from not-quite-ξt , by dividing on P(O|λ)

$$P(O|\lambda) = \sum_{j=1}^{N} \alpha_t(j)\beta_t(j)$$

Then final equation for ξt is

$$\xi_t(i,j) = \frac{\alpha_t(i)\, a_{ij} b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

Then expected number of transitions from state i to state j is then the sum over all t of ξ

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{k=1}^{N} \xi_t(i,k)}$$

And now try to estimate probability of a given symbol vk from the observation vocabulary V , given a state j

$$\hat{b}_j(v_k) = \frac{\text{expected number of times in state } j \text{ and observing symbol } v_k}{\text{expected number of times in state } j}$$

For this we need to compute

Calculate

$$\gamma_t(j) = P(q_t = j | O, \lambda)$$

Same way

$$\gamma_t(j) = \frac{P(q_t = j, O | \lambda)}{P(O|\lambda)}$$

$$\gamma_t(j) = \frac{\alpha_t(j)\beta_t(j)}{P(O|\lambda)}$$

$$\hat{b}_j(v_k) = \frac{\sum_{t=1\ s.t.O_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

function FORWARD-BACKWARD(*observations* of len *T*, *output vocabulary V*, *hidden state set Q*) **returns** *HMM=(A,B)*

initialize *A* and *B*
iterate until convergence
E-step

$$\gamma_t(j) = \frac{\alpha_t(j)\beta_t(j)}{\alpha_T(q_F)} \quad \forall\, t \text{ and } j$$

$$\xi_t(i,j) = \frac{\alpha_t(i)\, a_{ij} b_j(o_{t+1})\beta_{t+1}(j)}{\alpha_T(q_F)} \quad \forall\, t,\ i, \text{ and } j$$

M-step

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{k=1}^{N} \xi_t(i,k)}$$

$$\hat{b}_j(v_k) = \frac{\sum_{t=1 s.t.\ O_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

return *A*, *B*

```python
for i in range(10):
  print(train_data[i][:7])
```

```
[('Pierre', 'NOUN'), ('Vinken', 'NOUN'), (',', '.'), ('61', 'NUM'), ('years', 'NOUN'), ('old', 'ADJ'), (',', '.')]
[('Mr.', 'NOUN'), ('Vinken', 'NOUN'), ('is', 'VERB'), ('chairman', 'NOUN'), ('of', 'ADP'), ('Elsevier', 'NOUN'), ('N.V.', 'NOUN')]
[('Rudolph', 'NOUN'), ('Agnew', 'NOUN'), (',', '.'), ('55', 'NUM'), ('years', 'NOUN'), ('old', 'ADJ'), ('and', 'CONJ')]
[('A', 'DET'), ('form', 'NOUN'), ('of', 'ADP'), ('asbestos', 'NOUN'), ('once', 'ADV'), ('used', 'VERB'), ('*', 'X')]
[('The', 'DET'), ('asbestos', 'NOUN'), ('fiber', 'NOUN'), (',', '.'), ('crocidolite', 'NOUN'), (',', '.'), ('is', 'VERB')]
[('Lorillard', 'NOUN'), ('Inc.', 'NOUN'), (',', '.'), ('the', 'DET'), ('unit', 'NOUN'), ('of', 'ADP'), ('New', 'ADJ')]
[('Although', 'ADP'), ('preliminary', 'ADJ'), ('findings', 'NOUN'), ('were', 'VERB'), ('reported', 'VERB'), ('*-2', 'X'), ('more', 'ADV')]
[('A', 'DET'), ('Lorillard', 'NOUN'), ('spokewoman', 'NOUN'), ('said', 'VERB'), (',', '.'), ('``', '.'), ('This', 'DET')]
[('We', 'PRON'), ("'re", 'VERB'), ('talking', 'VERB'), ('about', 'ADP'), ('years', 'NOUN'), ('ago', 'ADP'), ('before', 'ADP')]
[('There', 'DET'), ('is', 'VERB'), ('no', 'DET'), ('asbestos', 'NOUN'), ('in', 'ADP'), ('our', 'PRON'), ('products', 'NOUN')]
```

```
Test Sentence: ['the', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog.']
Predicted Tags: ['DET', 'ADJ', 'NOUN', 'ADP', 'NOUN', 'ADP', 'DET', 'ADJ', 'NOUN']
Accuracy: 88.91%
```