Human-Computer Dialogue Simulation Using Hidden Markov Models

Presented by
Hamid R. Chinaei

March 12, 2008
Outline

- Language Modeling using HMM
- Tagging using HMM
- Viterbi Algorithm
- Simulation in SDS
- Simulation using HMM
- A Simulation Algorithm
Language Modeling

- We have some (finite) vocabulary, say
  \[ V = \{ \text{the, a, researve, book, flight, air, airline, \ldots} \} \]

- We have an (infinite) set of strings
  \[ V^* = \{ \text{the, a, the flight, book the flight, \ldots} \} \]

- We need to learn a probability distribution \( \hat{P} \) from a training sample with distribution \( P \)
  \[ \hat{P}(\text{the}) = 10^{-8} \]
  \[ \hat{P}(\text{the flight}) = 10^{-4} \]
Deriving a Trigram Probability Model

- Probability of "the flight booked"

\[ P(\text{the, flight, booked}) = P(\text{the}|\text{START}) \]
\[ \cdot p(\text{flight}|\text{START, the}) \]
\[ \cdot p(\text{booked}|\text{START, the, flight}) \]
\[ \cdot p(\text{STOP}|\text{START, the, flight, booked}) \]

- 3grams, general assumption:

\[ P(w_n|w_1, \ldots, w_{n-1}) = P(w_n|w_{n-1}, w_{n-2}) \]
Estimate Problem

- maximum likelihood

\[ p(\text{booked} | \text{the, flight}) = \frac{\text{Count(\text{the,flight,booked})}}{\text{Count(\text{the,flight})}} \]
Tagging

- **Part-of-speech tagging**
  - Input: The flight booked
  - OUTPUT: The/DT flight/NN booked/VB
    - DT: Determinant
    - NN: Noun
    - VB: Verb

- **Named entity tagging**
  - Input: An aircanada flight from Quebec to Toronto
  - OUTPUT: An aircanada/AIRLINE flight from Quebec/CITY to TORONTO/CITY
Hidden Markov Models for Tagging

- Sentence, $S = w_1, \ldots, w_n$
- Tag Sequence, $T = t_1, \ldots, t_n$
- Use an HMM to define:
  
  $P(t_1, \ldots, t_n, w_1, \ldots, w_n)$

- The most likely tag sequence for $S$ is:

  $$P(T, S) = P(\text{END} | t_{n-1}, t_n) \prod_{j=1}^{n} P(t_j | t_{j-2}, t_{j-1}) \prod_{j=1}^{n} P(w_j | t_j)$$

  $$T^* = \text{argmax}_T P(T, S)$$
Example

1. Sentence, The flight booked
2. Tag Sequence, DT NN VB

\[ P(T, S) = P(DT|START, START) \]
\[ \cdot P(NN|START, DT) \]
\[ \cdot P(VB|DT, NN) \]
\[ \cdot P(END|NN, VB) \]
\[ \cdot P(The|DT) \]
\[ \cdot P(flight|NN) \]
\[ \cdot P(booked|VB) \]
Smoothed Parameter Estimation

- Probability of generating booked/VB:
  \[ P(\text{booked}|\ VB) \cdot P(\ VB|\ DT, \ NN) \]

  \[
P(\ VB|\ DT, \ NN) = \gamma_1 \frac{\text{Count}(\ VB, \ DT, \ NN)}{\text{Count}(\ DT, \ NN)} \cdot \gamma_2 \frac{\text{Count}(\ NN, \ VB)}{\text{Count}(\ NN)} \cdot \gamma_3 \frac{\text{Count}(\ VB)}{\text{Count}()}\]

  \[
  \gamma_1 + \gamma_2 + \gamma_3 = 1 \text{ and } \gamma_i \geq 0
  \]

  \[
P(\text{booked}|\ VB) = \frac{\text{Count}(\ VB, \ booked)}{\text{Count}(\ VB)}
  \]
Viterbi Algorithm

- How to calculate $T^* = \arg\max_T \log P(T, S)$
- Define $n$ the length of the sentence
- Define a dynamic programming table
  $\pi[i, u, v] =$ maximum log probability of a tag sequence ending tags $u, v$ at position $i$
- We should calculate $\max_{u, v \in T} \pi[n, u, v]$
Viterbi Algorithm, cont’d

- Base case: \( \pi[0, \text{START}, \text{START}] = \log 1 = 0 \)
  \( \pi[0, u, v] = \log 0 = -\infty \)

- Recursive call for \( i = 1 \ldots n \), for all \( u, v \)

  \[
  \pi[i, u, v] = \max_{t \in T} \{ \pi[i - 1, t, u] + \text{Score}(S, i, t, u, v) \}
  \]

  where

  \[
  \text{Score}(S, i, t, u, v) = \log P(v|t, u) + \log P(w_i|v)
  \]

- Complexity \( O(nk^3) \), \( n \) is length of sentence and \( k \) is number of possible tags
Viterbi Algorithm: Running Time

- $O(n|T|^3)$ time to calculate $Score(S, i, t, u, v)$ for all $i, t, u, v$
- $n|T|^2$ entries in $\pi$ to be filled in
- $O(T)$ time to fill in one entry
- $\Rightarrow O(n|T|^3)$ time
SDS Dialogue Simulator
Conversations consist of system turns. In each turn the user’s responses are gathered.

- System turns are divided into sub-goals.

- Bigram language model used for predict of goal sequence given set of goals $G = \{g_1, \ldots, g_n\}$

$$P(g_n|g_1, \ldots, g_{n-1}) = P(g_n|g_{n-1})$$

- Language Model $\Lambda = (\sigma, \delta)$
  - $\sigma$ is the initial distribution and $\delta$ transitional distribution.
System Dynamics, cont’d

- $N$ states in a goal with a final state, can be modeled as a set of visible states $S = \{S_1, \ldots, S_N\}$
  - State at time $t$ is represented by $q_t$ and the final state $q_N$
- $M$ observed symbols, can be modeled as $V = \{v_1, \ldots, v_M\}$
  - Symbol observed at time $t$ is represented by $c_t$

- Flow of system turns $P(q_{t+1}|q_t)$
- Observation in each turn $P(c_t|q_t)$
- Start of conversation within a goal $\Pi = P(q_0)$
User intentions $H = h_1, \ldots, h_L$

- Intention at time $t$ is $u_t$

- Discrete random variable $U$ for user intentions generated in each state, $P(u_t | q_t, c_t)$
Hidden Markov Models

a) Multiple HMM connected by a bigram language model, $P(g_n|g_{n-1})$
b) HMM $\gamma = (A, B, \pi)$
c) IHMMs Input Hidden Markov Models $P(q_{t+1}|q_t, u_t)$
d) IOHMMs Input Hidden Markov Models $P(c_t|q_t, u_{t-1})$
Simulation Algorithm

01. function DialogueSimulator()
02. load parameters of the language model Λ
03. current_goal ← random goal from σ

04. while current_goal ≠ g_N do
05. λ ← parameters of the HMM given current_goal
06. SimulateHMM(λ)
07. current_goal ← random goal from δ
08. end
09. end

10. function SimulateHMM(λ)
11. t ← 0
12. q_t ← random system turn from π
13. c_t ← random system intention from P(c_t | q_t)

14. loop
15. print c_t
16. u_t ← random user intention from P(u_t | q_t, c_t)
17. print u_t
18. q_t ← random system turn from P(q_{t+1} | q_t)
19. if q_t = q_N then return
20. else t ← t + 1
21. c_t ← random system intention from P(c_t | q_t)
22. end
23. end
Use of language model and HMMs as a generator

- Given language model $\Lambda$, initial goal from $\sigma$ and goal transition from $\delta$ until finding the final goal $g_N$

- For each goal, the function Simulate HMM is invoked with corresponding HMM model $\lambda$ to generates sequence of system intentions $c_t$ and user intention $u_t$ until finding the final state $q_N$
Experimental Design

- Use of Communicator corpora 2001 annotated using DATE scheme
- Training using following 2 steps
  - Language model training
  - Dialogue segmentation, each segments corresponds to a goal
- HMM training
HMM Training

- Classification of system turns into states for the HMMs
  - System turns with speech acts
    \[ S = \{ \text{request - info, offer, acknowledgement} \} \]

- Classification of system turns into intentions
  - System turns have many combinations of dialogue acts, so divide into system intentions
    \[ V = \{ \text{start, apology, instruction, confirmation} \} \]

- request-info, offer, acknowledgement as start; implicit explicit-confirm as confirmation; apology as apology;

- Classification of user turns into intentions
  - \[ H = \{ \text{oov, command, yes, no, CITY, DATETIME, RENTAL, CAR, AIRLINE, HOTEL, AIRPORT, NUMBER, CITY CITY, DATE TIME DATE TIME, CITYDATE TIME, AIRLINEDATE TIME, AIRLINE NUMBER, CITY CITYDATE TIME} \} \]
Thanks!

Questions?