Markov Models

PA154 Language Modeling (4.1)

Pavel Rychlý

pary@fi.muni.cz

March 9, 2023

Source: Introduction to Natural Language Processing (600.465)
Jan Hajič, CS Dept., Johns Hopkins Univ.
www.cs.jhu.edu/˜hajic

Review: Markov Process

- Bayes formula (chain rule):

$$P(W) = P(w_1, w_2, .., w_T) = \Pi_{i=1..T} p(w_i \mid \cancel{w_1, w_2, ..}, w_{i-n+1}, .., w_{i-1})$$

- n-gram language models:
  - Markov process (chain) of the order n-1:

approximation

$$P(W) = P(w_1, w_2, .., w_T) = \Pi_{i=1..T} p(w_i \mid w_{i-n+1}, w_{i-n+2}, .., w_{i-1})$$

Using just <u>one</u> distribution (Ex.: trigram model: $p(w_i \mid w_{i-2}, w_{i-1})$) :

Positions: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Words: My car $\boxed{\textit{broke down}\ ,}$ and within hours Bob 's can $\boxed{\textit{broke down}\ ,}$ too .

p(, | broke down) = $p(w_5 \mid w_3, w_4) = p(w_{14} \mid w_{12}, w_{13})$

Markov Properties

- Generalize to any process (not just words/LM):
  - Sequence of random variables: $X = (X_1, X_2, \dots, X_T)$
  - Sample space S (*states*), size N: $S = (S_0, S_1, S_2, \dots, S_N)$

1. Limited History (Context, Horizon):

$\forall i \in 1..T; P(X_i \mid X_1, .., X_{i-1}) = (X_i \mid X_{i-1})$

1 7 3 7 9 0 6 7 $\boxed{3}$ 4 5... 1 7 3 7 9 0 6 7 $\boxed{3}$ 4 5...

2. Time invariance (M.C. is stationary, homogenous)

$\forall i \in 1..T, \forall y, x \in S; P(X_i = y \mid X_{i-1} = x) = p(y \mid x)$

1 $\boxed{7}$ $\boxed{3}$ $\boxed{7}$ $\boxed{9}$ 0 6 $\boxed{7}$ $\boxed{3}$ 4 5 ...

? ok...same ***distribution***

Long History Possible

- What if we want trigrams:

1 7 3 7 9 0 $\boxed{6\ 7}$ $\boxed{3}$ 4 5...

- Formally, use transformation:

Define new variables $Q_i, such that X_i = Q_{i-1}, Q_i$:

Then

$$P(X_i \mid X_{i-1}) = P(Q_{i-1}, Q_i \mid Q_{i-2}, Q_{i-1})$$

Predicting (X_i) 1 7 3 7 $\boxed{9\ 0}$ 6 7 3 4 5...

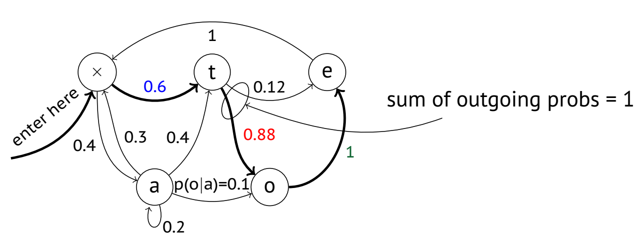
History ($X_i = \{Q_{i-2}, Q_{i-1}\}$):

x	1	7	3	...	0	6	7	3	4
x	x	1	7	...	9	0	6	7	3

Graph Representation: State Diagram

- S = $\{s_0, s_1, s_2, \dots, s_n\}$: states
- Distribution P($X_i \mid X_{i-1}$):
  - transitions (as arcs) with probabilities attached to them:

Bigram case:

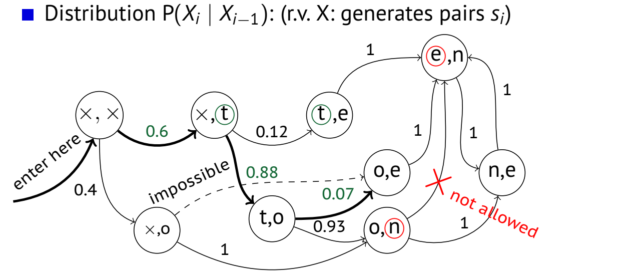


sum of outgoing probs = 1

p(toe) = .6 × .88 × 1 = .528

The Trigram Case

- S = $\{s_0, s_1, s_2, \dots, s_n\}$: states: pairs s_i = (x,y)
- Distribution P($X_i \mid X_{i-1}$): (r.v. X: generates pairs s_i)



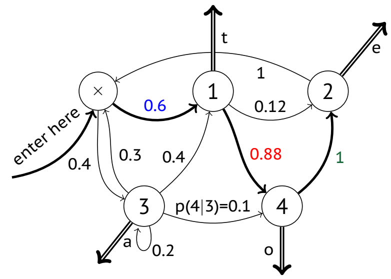
p(toe) = .6 × .88 × .07 ≅ .037 p(one) = ?

Finite State Automaton

- States $\sim$ symbols of the [input/output] alphabet
  - pairs (or more): last element of the n-tuple
- Arcs $\sim$ transitions (sequence of states)
- [Classical FSA: alphabet symbols on arcs:
  - transformation: arcs $\leftrightarrow$ nodes]
- Possible thanks to the "limited history" Markov Property
- So far: ***Visible*** Markov Models (VMM)

Hidden Markov Models

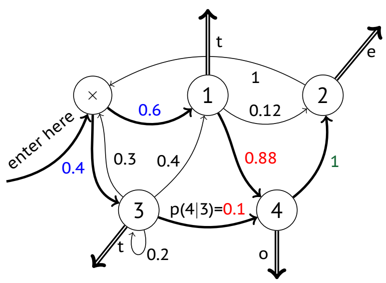
- The simplest HMM: states generate [observable] output (using the"data" alphabet) but remain "invisible":



p(toe) = .6 $\times$ .88 $\times$ 1 = .528

Added Flexibility...

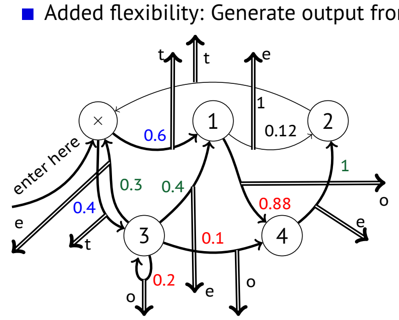
- So far, no change; but different states may generate the same output (why not?):



p(toe) = .6 $\times$ .88 $\times$ 1 +
.4 $\times$ .1 $\times$ 1 = .568

Output from Arcs...

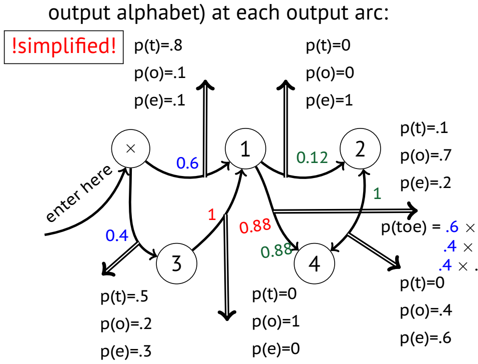
- Added flexibility: Generate output from arcs, not states:



p(toe) = .6 $\times$ .88 $\times$ 1 +
.4 $\times$ .1 $\times$ 1 +
.4 $\times$ .2 $\times$ .3 +
.4 $\times$ .2 $\times$ .4 = .624

...and Finally, Add Output Probabilities

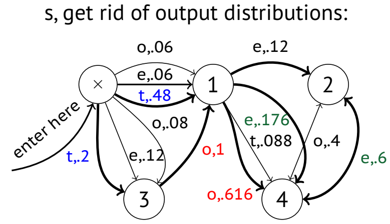
- Maximum flexibility: [Unigram] distribution (sample space: output alphabet) at each output arc:



p(toe) = .6 $\times$ .8 $\times$ .88 $\times$ .7 $\times$ 1 $\times$ .6 +
.4 $\times$ .5 $\times$ 1 $\times$ 1 $\times$ .88 $\times$ .2 +
.4 $\times$ .5 $\times$ 1 $\times$ 1 $\times$ .12 $\times$ 1
$\cong$ .237

Slightly Different View

- Allow for multiple arcs from $s_i \rightarrow s_j$, mark them by output symbol s, get rid of output distributions:



p(toe) = .48 $\times$ .616 $\times$ .6 +
.2 $\times$ 1 $\times$ .176 +
.2 $\times$ 1 $\times$ .12 $\cong$ .237

In the future, we will use the view more convenient for the problem at hand.

Formalization

HMM (the most general case):

- five-tuple (S, s_0, Y, P_S, P_Y), where:
  - S = $\{s_0, s_1, s_2, \ldots, s_T\}$ is the set of states, s_0 is the initial state,
  - Y = $\{y_1, y_2, \ldots, y_V\}$ is the output alphabet,
  - $P_S(s_j \mid s_i)$ is the set of prob. distributions of transitions,
    - size of P_s :$\mid S \mid^2$.
  - $P_Y(y_k \mid s_i, s_j)$ is the set of output (emission) probability distributions.
    - size of P_Y :$\mid S \mid^2 x \mid Y \mid$

Example:

- S= x, 1, 2, 3, 4, s_0 = x
- Y = $\{t, o, e\}$

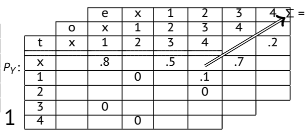
Formalization - Example

- Example (for graph, see foils 11,12):
  - S= $\{x, 1, 2, 3, 4\}, s_0$ = x
  - Y= $\{e, o, t\}$
  - P_S :

	x	1	2	3	4
x	0	.6	0	.4	0
1	0	0	.12	0	.88
2	0	0	0	0	1
3	0	1	0	0	0
4	0	0	1	0	0

$\longrightarrow \Sigma = 1$

P_Y:



Using the HMM

- The generation algorithm (of limited value :-)):
  1. Start in $s = s_0$.
  2. Move from s to s' with probability $P_S(s' \mid s)$.
  3. Output (emit) symbol y_k with probability $P_Y(y_k \mid s, s')$.
  4. Repeat from step 2 (until somebody says enough).
- More interestirig usage:
  - Given an output sequence $Y = \{y_1, y_2, \ldots, y_k\}$ compute its probability.
  - Given an output sequence $Y = \{y_1, y_2, \ldots, y_k\}$ compute the most likely sequence of states which has generated it.
  - ...plus variations: e.g., n best state sequences