Natural Language Processing

2. Probabilistic language models

Katya Chernyak

CS HSE

September 11, 2018

Overview

1. Language models
2. Hidden Markov Models (HMM)
3. Maximum Entropy Markov Models (MEMM)
4. Conditional Random Field (CRF)

Language model

1. Compute the probability of a sequence of words:

$$P(w_1, w_2, \dots, w_n)$$

2. Predict next word:

$$P(w_n | w_1, w_2, \dots, w_{n-1})$$

LMs help are used to:

- Machine translation: choose best translation
- Spell checking: find incorrect word
- Speech recognition: choose best transcription
- Predict next word in your smartphone
- Generate poems, summaries, answers, etc.

А. А. Марков (1886).

Markov assumptions

1. Chain rule:

$$P(W) = P(w_1, w_2, \dots, w_n) = \prod_i P(w_i | w_1, \dots, w_{i-1})$$

2. Maximum likelihood estimates of probabilities:

$$P(w_i | w_1, \dots, w_{i-1}) = \frac{\text{count}(w_1, w_2, \dots, w_i)}{\text{count}(w_1, \dots, w_{i-1})}$$

3. Markov assumption (k-th order):

$$P(w_i | w_1, \dots, w_{i-1}) \approx P(w_i | w_{i-k}, \dots w_{i-1})$$

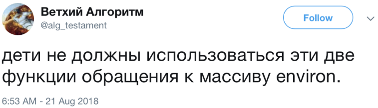
n-gram models

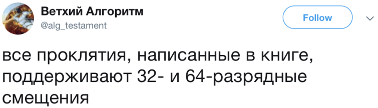
1. Unigram models: $P(W) = P(w_1, \ldots, w_n) \approx \prod_i P(w_i)$
2. Bigram models: $P(W) = P(w_1, \ldots, w_n) \approx \prod_i P(w_i|w_{i-1})$
3. Perplexity: $PP(W) = P(w_1, w_2, \ldots w_n)^{-\frac{1}{N}}$
   The lower perplexity, the better the model predicts an unseen test
4. Smoothing: $P(w_i|w_1, \ldots, w_{i-1}) = \frac{\text{count}(w_1,w_2,\ldots,w_i)+1}{\text{count}(w_1,\ldots,w_{i-1})+\alpha|V|}$, where $|V|$ is the size of dictionary
5. Interpolation: $\hat{P}(w_i|w_{i-1}) = \lambda P_{MLE}(w_i|w_{i-1}) + (1-\lambda)P_{MLE}(w_i)$

n-gram models for text generation

Given w_i:

1. choose the next most probable w_{i+i}
2. randomly select sample from this probability distribution of next words





`https://twitter.com/alg_testament`

n-gram models for IR

Given documents D and query q, estimate the probability of generating the query text from a document language model:

- Rank documents by the probability that the query could be generated by the document model;
- Calculate $P(d|q)$ to rank the documents: $P(d|q) \propto P(q|d)P(d)$
- Assuming prior is uniform, unigram model: $P(q|d) = \prod_i P(q_i|d)$
- MLE: $P(q_i|d) = \frac{\text{count}(q_i,d)}{|d|}$

Overview

1. Language models
2. Hidden Markov Models (HMM)
3. Maximum Entropy Markov Models (MEMM)
4. Conditional Random Field (CRF)

Part of speech tagging

Given a sentence or a sequence of words (X), predict its part of speech sequence (Y)

X (words)	the	cat	sat	on	a	mat
Y (POS-tags):	DET	NOUN	VERB	PREP	DET	NOUN

1. Pointwise prediction: choose a POS-tag for a word individually
2. Sequence models:
   - Generative models: $P(y, x)$
   - Discriminative models: $P(y|x)$

Generative sequence models

$$\arg\max_{Y} P(Y|X) = \arg\max \frac{P(X|Y)P(Y)}{P(X)} \approx \arg\max P(X|Y)P(Y)$$

- $P(X|Y)$ models word/ P OS tag interactions
- $P(Y)$ models POS / POS interactions

А. А. Марков (1886).

Hidden Markov Models

An HMM is specified by the following components:

$Q = q_1, \ldots, q_T$	states (POS-tags)
$A = (a_{ij})$	transition probability matrix: $a_{ij} = P(Q_i \to Q_j)$
$O = o_1, \ldots, o_V$	observations (words)
B	emission probabilities
	$b_i(o_t)$ is the probability of q_i generate o_t
$\pi = \pi_1, \ldots, \pi_N$	initial probability distribution

Probabilities should sum to unity:

$\sum_j a_{ij} = 1$

$\sum_i \pi_i = 1$

Markov assumptions

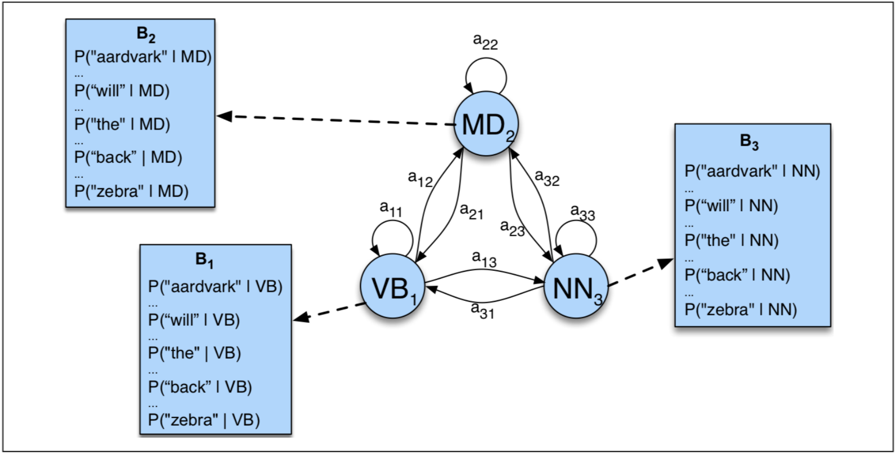
1. The probability of a particular state depends only on the previous state:

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

2. Output Independence: the probability of an output observation o_i depends only on the state that produced the observation q_i:

$$P(o_i|Q,O) = P(o_i|q_i)$$

Example of HMM (from SLP Book [1])



[1] `https://web.stanford.edu/~jurafsky/slp3`

Three tasks of HMM

1. Likelihood: given an observation sequence, estimate the likelihood of the observation sequence
2. Decoding: given an observation sequence, discover the best hidden state sequence leading to these observations.
3. Learning: train HMM

Forward-backward algorithm

$O_n = o_1, \dots, o_n$
Forward probabilities: $\alpha_{ij} = P(o_1, \dots, o_i)$
Forward algorithm:

1. $\alpha_{1j} = a_{0j} b_j(o_1), 1 \leq j \leq |Q|$
2. $\alpha_{ij} = \sum_k^Q \alpha_{i-1,k} a_{kj} b_j(o_i), 1 \leq i \leq n, 1 \leq j \leq |Q|$
3. $P(O_n) = \sum_k^Q \alpha_{nk} a_{kF}$

Backward probabilities: $beta_{oj} = P(o_{i+1}, \dots, o_n)$
Backward algorithm:

1. $\beta_{nj} = a_{jF}, 1 \leq j \leq |Q|$
2. $\alpha_{ij} = \sum_k^Q \beta_{i+1,k} a_{jk} b_k(o_{i+1}), 1 \leq i \leq n, 1 \leq j \leq |Q|$
3. $P(O_n) = \sum_k^Q a_{0k} b_(o_1) \beta$

Decoding

Input: HMM $= (A, B)$, observations $= o_1, \ldots, o_n$
Output: the most probable sequence of states $= q_1, \ldots, q_n$

$$\hat{q}_n = \arg\max_{q_n} P(q_n|o_n) \approx$$

$$\approx \max_{q_n} \prod_{i=1}^{n} P(o_i|q_i) P(q_i|q_{i-1})$$

Viterbi algorithm

Compute path probabilities $V = |n \times T|$. v_{ij} represents the probability that the HMM is in state j after seeing the first i observations.

1. Intialize

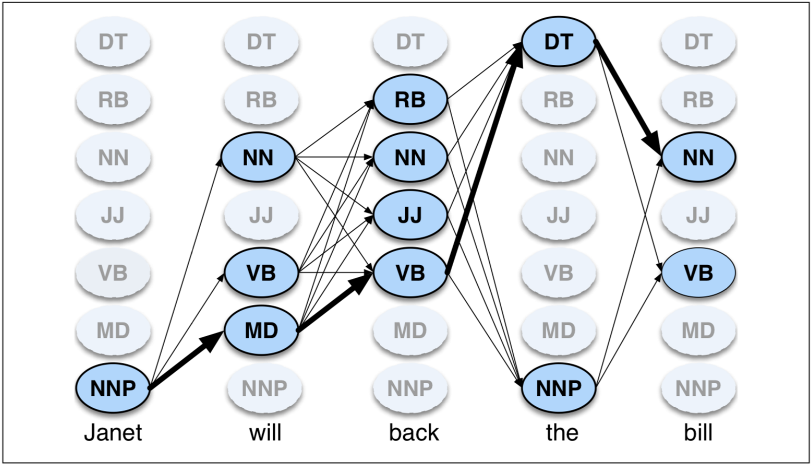
$$v_{1j} = a_{0j} b(o_1), 1 \le j \le T$$

2. Recursion

$$v_{ij} = \max v_{i-1,k} a_{kj} b_j(o_i), 1 \le i \le n, 1 \le j \le T$$

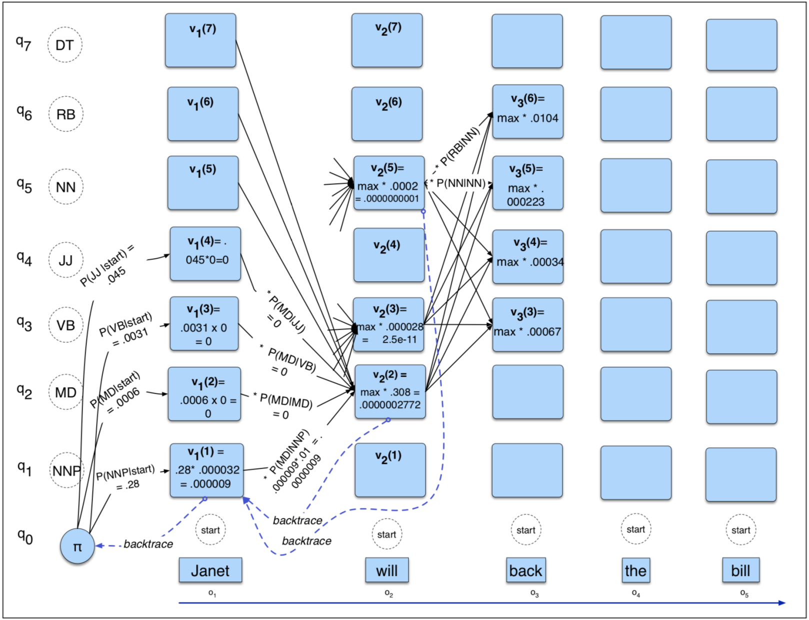
3. End

$$\max_{q \in Q^n} p(o, q) = \max_{1 \le k \le T} v_{nk} a_{kF}$$


DT
RB
NN
JJ
VB
MD
NNP
Janet
will
back
the
bill

	Janet	will	back	the	bill
NNP	0.000032	0	0	0.000048	0
MD	0	0.308431	0	0	0
VB	0	0.000028	0.000672	0	0.000028
JJ	0	0	0.000340	0	0
NN	0	0.000200	0.000223	0	0.002337
RB	0	0	0.010446	0	0
DT	0	0	0	0.506099	0

	NNP	MD	VB	JJ	NN	RB	DT
$<s>$	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	0.0036
NN	0.0096	0.0176	0.0014	0.0086	0.1216	0.0177	0.0068
RB	0.0068	0.0102	0.1011	0.1012	0.0120	0.0728	0.0479
DT	0.1147	0.0021	0.0002	0.2157	0.4744	0.0102	0.0017



q7 DT
q6 RB
q5 NN
q4 JJ
q3 VB
q2 MD
q1 NNP
q0 π
v1(7)
v1(6)
v1(5)
v1(4)= .045*0=0
v1(3)= .0031 x 0 = 0
v1(2)= .0006 x 0 = 0
v1(1) = .28* .000032 = .000009
v2(7)
v2(6)
v2(5)= max * .0002 = .0000000001
v2(4)
v2(3)= max * .000028 = 2.5e-11
v2(2) = max * .308 = .0000002772
v2(1)
v3(6)= max * .0104
v3(5)= max * .000223
v3(4)= max * .00034
v3(3)= max * .00067
P(JJ|start) = .045
P(VB|start) = .0031
P(MD|start) = .0006
P(NNP|start) = .28
* P(MD|JJ) = 0
* P(MD|VB) = 0
* P(MD|MD) = 0
* P(MD|NNP) .000009*.01 = .0000009
* P(RB|NN)
* P(NN|NN)
backtrace
backtrace
backtrace
start
start
start
start
start
Janet
will
back
the
bill
o1
o2
o3
o4
o5

TnT POS-tagger [Brants, 2000]

TnT uses second-order HMM for POS-tagging:

$$\arg\max[\prod_{j}[p(t_i|t_{i-1}, o_{t-2})p(w_i|t_i)]P(t_{T+1}|t_T)$$

The probability of a POS-tag for a given word is computed as a linear interpolation of three LM's:

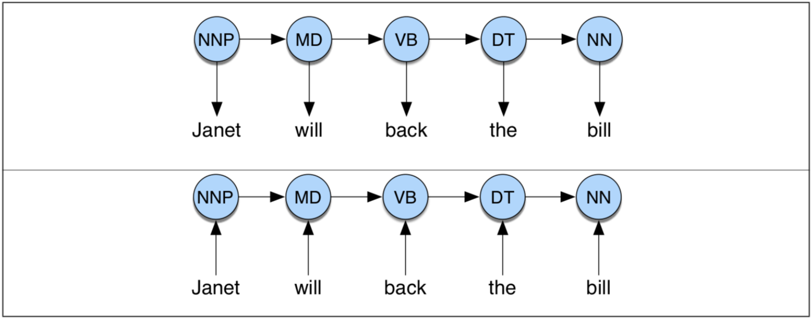
$$P(t_i|t_{i-1}, t_{i-2}) = l_1 * P(t_i) + l_2 * P(t_i|t_{i-1}) + l_3 * P(t_i|t_{i-1}, t_{i-2})$$

See NLTK examples (English_HMM_POS_tagger).

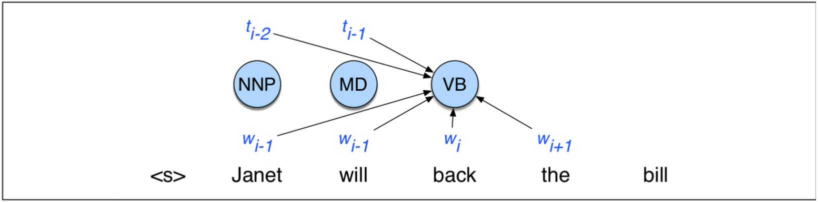
Overview

Maximum Entropy Markov Models (MEMM)

HMM: $\arg\max P(Y|X) = \arg\max_Y P(X|Y)P(Y)$
MEMM: $\arg\max_y P(Y|X) = \arg\max_Y P(y_i|y_{i-1}, x_i)$



Features in a MEMM



- Feature templates: $< t_i, w_{i-2} >$, $< t_i, t_{i-1} >$, $< t_i, t_i, w_i, w_{i+1} >$
- Casing, shape, is number?, is string?, has a dash?, has a digit?, etc.

Decoding MEMM

- Locally normalized logistic regression on a sequencer:

$$\hat{Y} = \arg\max P(Y|X) = \arg\max \prod_i P(t_i, w_{i-l}^{i+l}, t_{i-k}^{i-1}) =$$

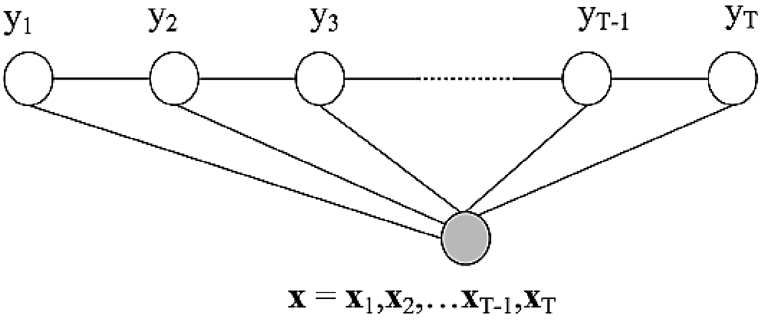
$$= \arg\max_T \prod_i \frac{exp(\sum_j \theta_j f_j(t_i, w_{i-l}^{i+l}, t_{i-k}^{i-1}))}{\sum_{t' \in T exp(\sum_j \theta_j f_j(t'_i, w_{i-l}^{i+l}, t_{i-k}^{i-1}))}}$$

- Viterbi recursion step: $v_{ij} = \max_k v_{i-1,k} P(t_k | t_{k-1}, w_i)$
- Local normalization leads to labels bias: will/NN to/TO fight/VB

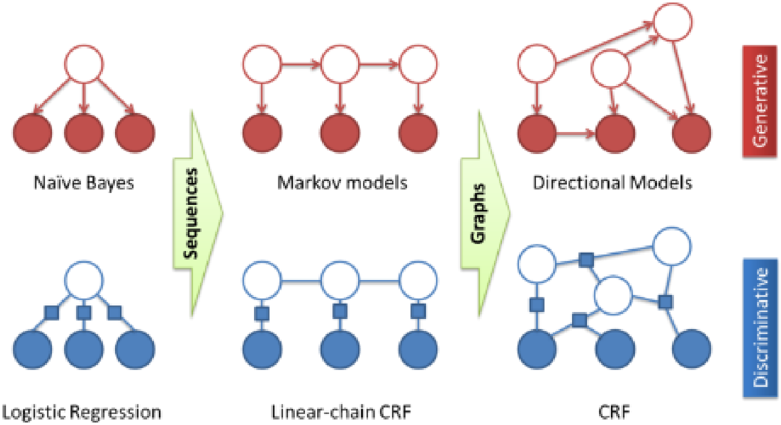
Overview

Conditional Random Field (CRF)

$$p(Y|X) = \frac{e^{\sum_{i=1}^{k} \lambda_i F_i(y,x)}}{\sum_{y' \in C^n} e^{\sum_{i=1}^{k} \lambda_i F_i(y',x)}}$$



HMM VS CRF



Adapted from C. Sutton, A. McCallum, "An Introduction to Conditional Random Fields", ArXiv, November 2010

Take aways

1. POS-tagging is **sequence labelling** task
2. HMMs and CRFs are a generative-discriminative pair
3. MEMM suffer from label bias problem and is rarely used
4. CRF is basically sequential logistic regression
5. Say hi to Andrew Viterbi <3!

What is next?

1. Neural language models
2. RNNs and CRFs are best friends
3. Probabilistic context-free grammars (PCFG) and CYK

Reading

1. Sutton, C. An Introduction to Conditional Random Fields. 2012
2. Stuart Russell, Peter Norvig. Artificial Intelligence: A Modern Approach, Ch. 15
3. Dan Jurafsky, James H. Martin. Speech and Language Processing, Ch. 3, Ch. 8