Natural Language Processing
CSE 325/425

Sihong Xie

Lecture 10:
- Maximum entropy principle
- Maximum entropy Markov model

Maximum entropy principle

Logistic regression models is a special case of the Maximum Entropy models.

What's the "best" estimation of a prob. distribution with partial information?

- Estimate the probabilities of seeing each of the six faces of a dice: $= \{1, 2, 3, 4, 5, 6\}$
  - Without any information, the best estimation is $\boxed{\Pr(i) = 1/6}$.
    - Entropy
$$H(X) = -\sum_{i=1}^{6} \Pr(i) \log \Pr(i)$$
    - For discrete random variables, maximum entropy <=> uniform distributions.

"$\Rightarrow$" don't over commit to any option.

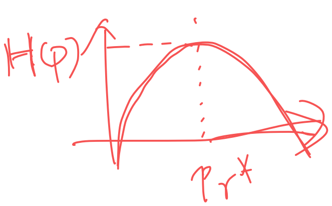
Extra Information (prior knowledge)

  - If we know that the odd numbers are twice more likely than ~~odd~~ even numbers, then
$$\Pr(1) = \Pr(3) = \Pr(5) = 2/9 \qquad \Pr(2) = \Pr(4) = \Pr(6) = 1/9$$
    - Still have maximum entropy, but also conform to the constraints (what are they?)

$\max_{\Pr} H(\Pr)$ $\leftarrow$ concave

constraints
$$\begin{cases} \Pr(1) + \Pr(3) + \Pr(5) = 2\left[\Pr(2) + \Pr(4) + \Pr(6)\right] \\ 1 \geq \Pr(i) \geq 0, \quad i = 1, \ldots, 6 \\ \sum_{i=1}^{6} \Pr(i) = 1 \end{cases}$$

$\leftarrow$ linear constraints



Maximum entropy principle

Predict the POS-tag of the word "*zzfish*".

- Without any information, we give equal probabilities to all tags.
- More information: "*zzfish*" can only be tagged as one of {NN, JJ, NNS, VB}:

cat
dog
human

plural

cats
dogs
humans

$$\Pr(NN) = \Pr(JJ) = \Pr(NNS) = \Pr(VB) = 1/4$$

- More information: "*zzfish*" is a sort of noun in 8 out of 10 times:

$$\Pr(NN) = \Pr(NNS) = 2/5 \qquad \Pr(JJ) = \Pr(VB) = 1/10$$

- More information: "*zzfish*" is a verb in 1 out of 20 times:

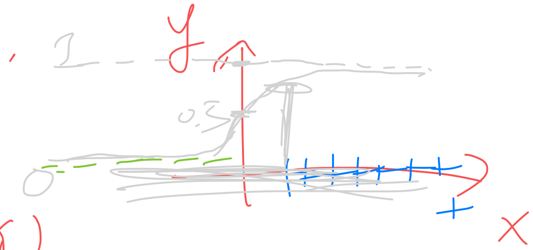
$$\Pr(NN) = \Pr(NNS) = 2/5 \qquad \Pr(JJ) = 3/20 \qquad \Pr(VB) = 1/20$$

$$1 \geq \Pr(\text{tag}) \geq 0$$

$$\sum_{c} \Pr(c) = 1$$

Setting: Training data $(x^{(i)}, y^{(i)})$ $i=1,\dots,m$,

$x^{(i)} \in \mathbb{R}$, $y^{(i)} \in \{0, 1\}$



train a logistic Regression model that outputs $\Pr(y=1|x)$

Maximum entropy principle

Logistic regression is a maximum entropy classifier

- Go through the last HW question of CSE326/426.

Warning: we're not using the traditional MLE to train the LR model

maximize the entropy of the random variable Y

$X \longrightarrow Y \in \{0,1\}$

$$\max \; H(Y) = -\sum_{i=1}^{m} \sum_{y=0}^{1} \Pr(y^{(i)}|x^{(i)}) \log \Pr(y^{(i)}|x^{(i)})$$

[Total entropy of $Y^{(1)} \dots Y^{(m)}$]

$$\text{s.t.} \quad \sum_{y=0}^{1} \Pr(y^{(i)} = y | x^{(i)}) = 1, \quad i = 1 \dots m$$

$$0 \le \Pr(y^{(i)} = y | x^{(i)}) \le 1, \quad \text{for all } i = 1, \dots m, \text{ all } y = 0, 1$$

$$\mathbb{E}_{\boxed{\Pr}}[X] = \sum_{i=1}^{m} \Pr(y | x^{(i)}) \, x^{(i)} \qquad \forall y = 0, 1$$

Training

$$= \sum_{i=1}^{m} [y^{(i)} = y] \, x^{(i)}$$

$$= \mathbb{E}_{\text{observed}}[X] \quad (\text{Empirical distribution})$$

Maximum entropy principle

Logistic regression is a maximum entropy classifier

- Go through the last HW question of CSE326/426.

Maximum Entropy distribution over y given some x

$$\boxed{\Pr(y|x) = \frac{e^{\lambda(y)x}}{\sum_{y'=0}^{1} e^{\lambda(y')x}}}$$

$\Rightarrow \Pr(y=1 \mid x)$

$$= \frac{1}{1+\exp(-\lambda(1)x)}$$

(sigmoid function)

where $\lambda(y) \in \mathbb{R}$ for class $y \in \{0,1\}$

① parameter for class y in LR

② Lagrangian multiplier of the constraint

$$\mathbb{E}_{\Pr}[X] = \mathbb{E}_{\text{empirical}}[X]$$

$Z(q_{t-1}, o_t)$

$= \sum_c \exp\{\langle \theta^c, \vec{f}(q_{t-1}, \boxed{q_t = c}, o_t) \rangle\}$ = a function of (q_{t-1}, o_t)

Maximum entropy Markov model

Not MEMM

Multi-class logistic regression can predict a POS-tag for one word, then predict the POS-tag q_t using the *fixed* previous predicted tag q_{t-1} . HMM ≈

$c \in \{1, \dots, N\}$

$$\Pr(q_t = c | q_{t-1}, o_t; \boldsymbol{\theta}^c) = \frac{1}{\underbrace{Z(q_{t-1}, o_t)}} \exp\left\{ \sum_{i=1}^{d} \theta_i^c \underbrace{f_i(q_{t-1}, o_t, q_t = c)} \right\}$$

"Prob of transiting from $q_{t-1} \to q_t = c$ & observed word o_t"

In HMM: $P(q_t | q_{t-1}) * P(o_t | q_t)$

Secretariat	is	expected	to	race	tomorrow.
NNP →	VBZ →	VBN →	TO →	VB →	RB
	NN	RB		NN	

- Pro: simple and surprisingly good results. TO
- Cons:
  - Can't use later predictions to correct previous predictions.
  - Errors propagate.

$f(q_{t-1} = \text{VBN}, q_t = \text{TO}, o_t = \text{"to"}) \in \{0, 1\}^d$

Maximum entropy Markov model

Marry logistic regression and Markov model

- Predict the whole sequence of POS-tags via Viterbi algorithm.
- When predicting a tag, the decision is made based on information from both directions.
  - Slightly modify the multi-class logistic regression model to predict q_t based on all possible q_{t-1}.

\# of All POS-tags

vary q_{t-1} from 1 to N

no longer fixed

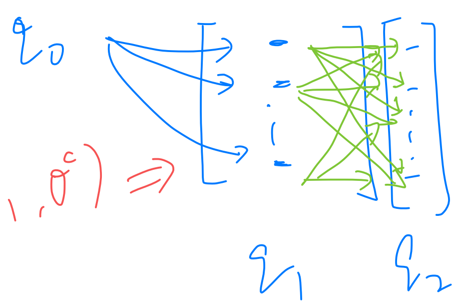
$$a_{ij}\ b_j(o_t) \Leftrightarrow \Pr(q_t = c|q_{t-1}, o_t; \boldsymbol{\theta}^c) = \frac{1}{Z(q_{t-1}, o_t)} \exp\left\{\sum_{i=1}^{d} \theta_i^c f_i(\boxed{q_{t-1}}, o_t, q_t = c)\right\}$$

dummy

Secretariat is expected to race tomorrow.

vary both c', c

destination



$\Pr(q_1 = c|q_0, o_1, \theta^c) \Rightarrow$

$\Leftarrow \Pr(q_2 = c|q_1 = c', o_2, \theta^c)$ = "compatibility of c', c, & o^t given θ^c"

source

Predict optimal sequences

Viterbi algorithm (for MEMM): compute maximum probability and the optimal sequence

1. Initialize $v_1(i)$ for each value i of the first hidden state q_1 .
2. for $t = 2, \dots, T$

   for $j = 1, \dots, N$

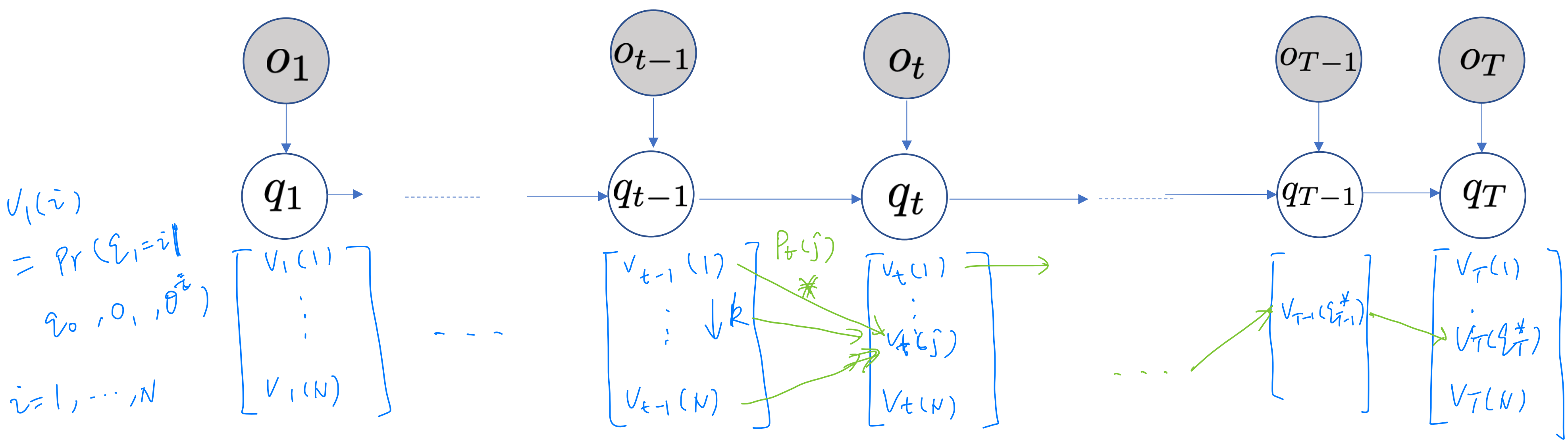
   HMM

   compute $v_t(j) = \cancel{\max_k v_{t-1}(k) \boxed{a_{kj} b_j(o_t)}}$

   record back-pointers $p_t(j) = \arg\max_k \cancel{v_{t-1}(k) a_{kj} b_j(o_t)}$

   MEMM $v_t(j) = \max_{k=1,\dots,N} v_{t-1}(k) \boxed{\Pr(q_t = j \mid q_{t-1} = k, o_t, \vec{\theta}^j)}$

   $v_{t-1}(k) \Pr(q_t = j \mid q_{t-1} = k, o_t, \vec{\theta}^j)$
3. Backtracking to find Q^*
4. Return $\Pr(Q^*|O) = \max_j v_T(j)$



Maximum entropy Markov model

Training of MEMM

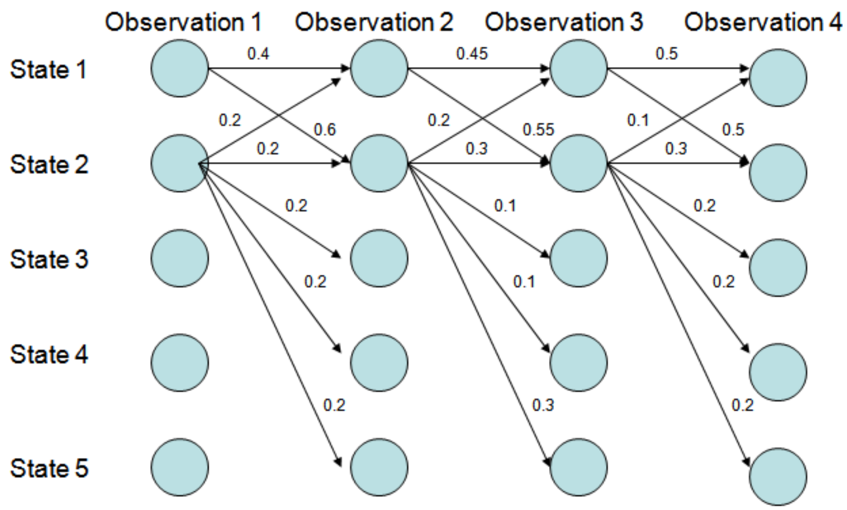
- Can only train with some labeled data (supervised or semi-supervised learning).
- Design feature functions $\mathbf{f}(q_{t-1}, o_t, q_t = c)$ for all possible tags and words.
- Evaluate the feature functions on the POS-tag sentences.
  - Go through each word, based on the observed word, the previous and the current POS-tags, compute the features.
  - Usually need to consider rich features beyond the word and tag identities.
    - consider suffixes, prefixes, Captipalizations, lower-cases, hyphens, dictionaries, tags that are farther away.
- Parameters: $\boldsymbol{\theta}^c, c \in \{\text{All POS-tags}\}$

Maximum entropy Markov model

Drawbacks of MEMM:

- Labeling bias in the tag probabilities, due to local normalization

$$\Pr(q_t = c | q_{t-1}, o_t; \boldsymbol{\theta}^c) = \frac{1}{Z(q_{t-1}, o_t)} \exp\left\{ \sum_{i=1}^{d} \theta_i^c f_i(q_{t-1}, o_t, q_t = c) \right\}$$



1 tends to move to 2 then stay with 2:

$$\Pr(1 \to 1 \to 1 \to 1) =$$

$$\Pr(1 \to 2 \to 2 \to 2) =$$

Next lecture:
address this issue using Conditional Random Fields