
CS 388:
Natural Language Processing:
Part-Of-Speech Tagging,
Sequence Labeling, and
Hidden Markov Models (HMMs)

Raymond J. Mooney
University of Texas at Austin

1

Part Of Speech Tagging

- Annotate each word in a sentence with a part-of-speech marker.
- Lowest level of syntactic analysis.

John saw the saw and decided to take it to the table.
NNP VBD DT NN CC VBD TO VB PRP IN DT NN

- Useful for subsequent syntactic parsing and word sense disambiguation.

2

English POS Tagsets

- Original Brown corpus used a large set of 87 POS tags.
- Most common in NLP today is the Penn Treebank set of 45 tags.
 - Tagset used in these slides.
 - Reduced from the Brown set for use in the context of a parsed corpus (i.e. treebank).
- The C5 tagset used for the British National Corpus (BNC) has 61 tags.

3

English Parts of Speech

- Noun (person, place or thing)
 - Singular (NN): dog, fork
 - Plural (NNS): dogs, forks
 - Proper (NNP, NNPS): John, Springfields
 - Personal pronoun (PRP): I, you, he, she, it
 - Wh-pronoun (WP): who, what
- Verb (actions and processes)
 - Base, infinitive (VB): eat
 - Past tense (VBD): ate
 - Gerund (VBG): eating
 - Past participle (VBN): eaten
 - Non 3rd person singular present tense (VBP): eat
 - 3rd person singular present tense (VBZ): eats
 - Modal (MD): should, can
 - To (TO): to (to eat)

4

English Parts of Speech (cont.)

- Adjective (modify nouns)
 - Basic (JJ): red, tall
 - Comparative (JJR): redder, taller
 - Superlative (JJS): reddest, tallest
- Adverb (modify verbs)
 - Basic (RB): quickly
 - Comparative (RBR): quicker
 - Superlative (RBS): quickest
- Preposition (IN): on, in, by, to, with
- Determiner:
 - Basic (DT) a, an, the
 - WH-determiner (WDT): which, that
- Coordinating Conjunction (CC): and, but, or,
- Particle (RP): off (took off), up (put up)

5

Closed vs. Open Class

- **Closed class** categories are composed of a small, fixed set of grammatical function words for a given language.
 - Pronouns, Prepositions, Modals, Determiners, Particles, Conjunctions
- Open class categories have large number of words and new ones are easily invented.
 - Nouns (Googler, textlish), Verbs (Google), Adjectives (geeky), Adverb (automagically)

6

Ambiguity in POS Tagging

- “Like” can be a verb or a preposition
 - I like/VBP candy.
 - Time flies like/IN an arrow.
- “Around” can be a preposition, particle, or adverb
 - I bought it at the shop around/IN the corner.
 - I never got around/RP to getting a car.
 - A new Prius costs around/RB \$25K.

7

POS Tagging Process

- Usually assume a separate initial tokenization process that separates and/or disambiguates punctuation, including detecting sentence boundaries.
- Degree of ambiguity in English (based on Brown corpus)
 - 11.5% of word types are ambiguous.
 - 40% of word tokens are ambiguous.
- Average POS tagging disagreement amongst expert human judges for the Penn treebank was 3.5%
 - Based on correcting the output of an initial automated tagger, which was deemed to be more accurate than tagging from scratch.
- Baseline: Picking the most frequent tag for each specific word type gives about 90% accuracy
 - 93.7% if use model for unknown words for Penn Treebank tagset.

8

POS Tagging Approaches

- **Rule-Based:** Human crafted rules based on lexical and other linguistic knowledge.
- **Learning-Based:** Trained on human annotated corpora like the Penn Treebank.
 - **Statistical models:** Hidden Markov Model (HMM), Maximum Entropy Markov Model (MEMM), Conditional Random Field (CRF)
 - **Rule learning:** Transformation Based Learning (TBL)
 - **Neural networks:** Recurrent networks like Long Short Term Memory (LSTMs)
- Generally, learning-based approaches have been found to be more effective overall, taking into account the total amount of human expertise and effort involved.

9

Problems with Sequence Labeling as Classification

- Not easy to integrate information from category of tokens on both sides.
- Difficult to propagate uncertainty between decisions and “collectively” determine the most likely joint assignment of categories to all of the tokens in a sequence.

10

Probabilistic Sequence Models

- Probabilistic sequence models allow integrating uncertainty over multiple, interdependent classifications and collectively determine the most likely global assignment.
- Two standard models
 - Hidden Markov Model (HMM)
 - Conditional Random Field (CRF)

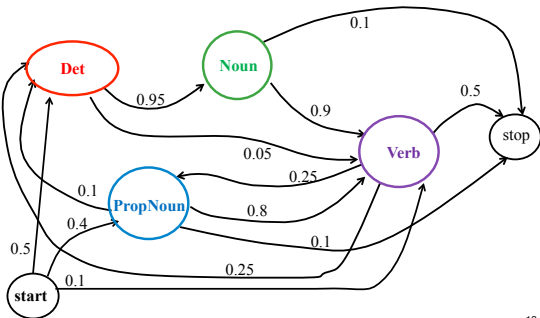
11

Markov Model / Markov Chain

- A finite state machine with probabilistic state transitions.
- Makes Markov assumption that next state only depends on the current state and independent of previous history.

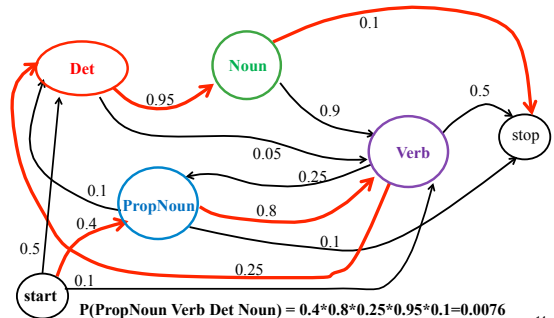
12

Sample Markov Model for POS



13

Sample Markov Model for POS



$$P(\text{PropNoun Verb Det Noun}) = 0.4 * 0.8 * 0.25 * 0.95 * 0.1 = 0.0076$$

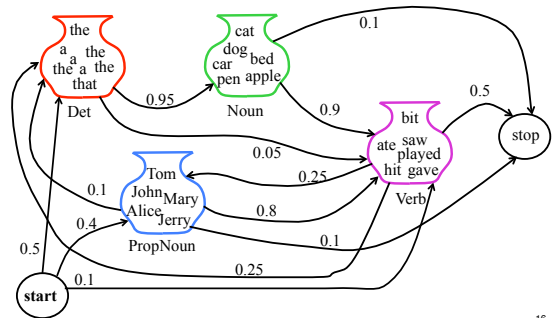
14

Hidden Markov Model

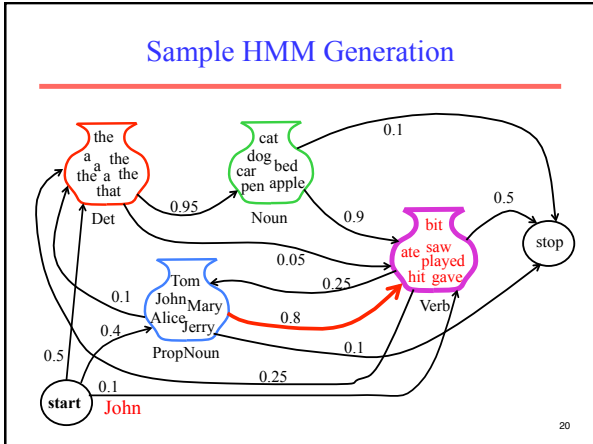
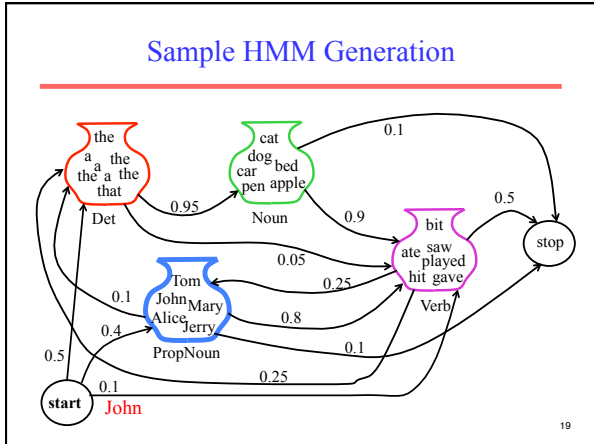
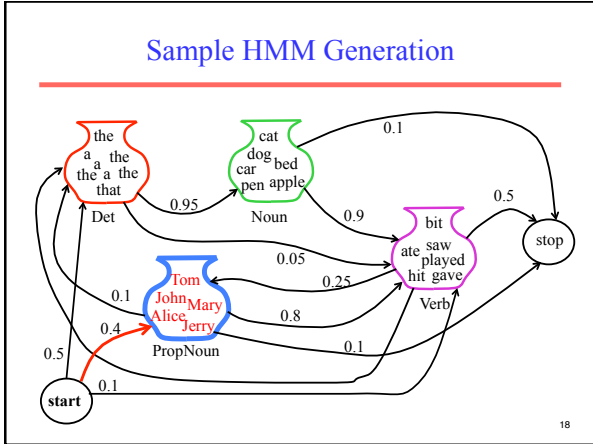
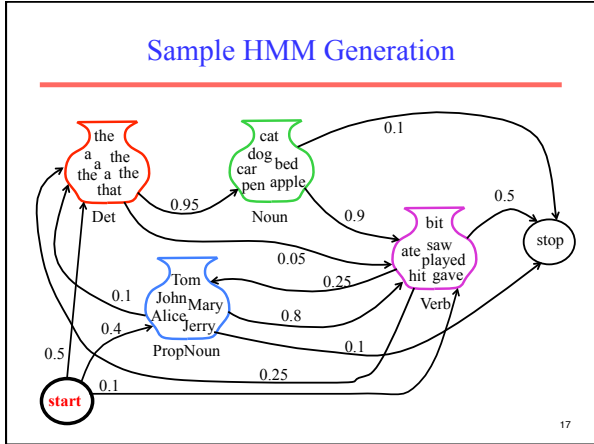
- Probabilistic generative model for sequences.
- Assume an underlying set of *hidden* (unobserved, latent) states in which the model can be (e.g. parts of speech).
- Assume probabilistic transitions between states over time (e.g. transition from POS to another POS as sequence is generated).
- Assume a *probabilistic* generation of tokens from states (e.g. words generated for each POS).

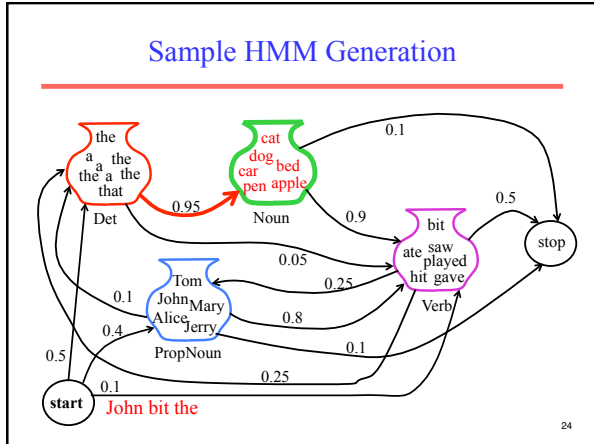
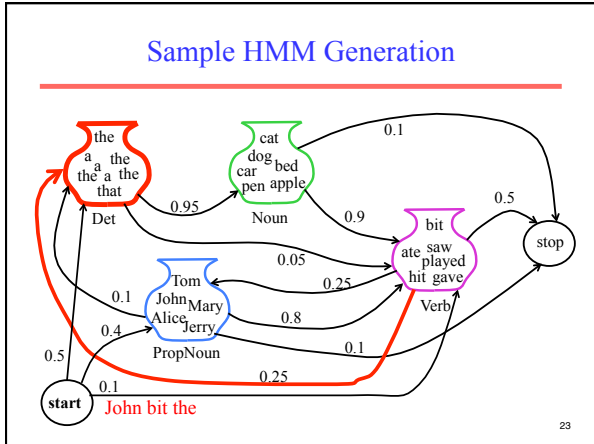
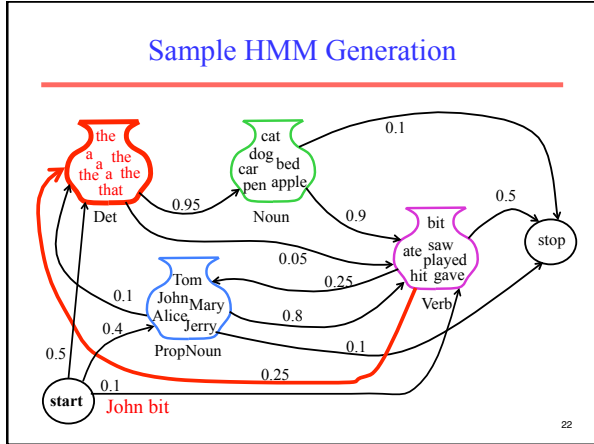
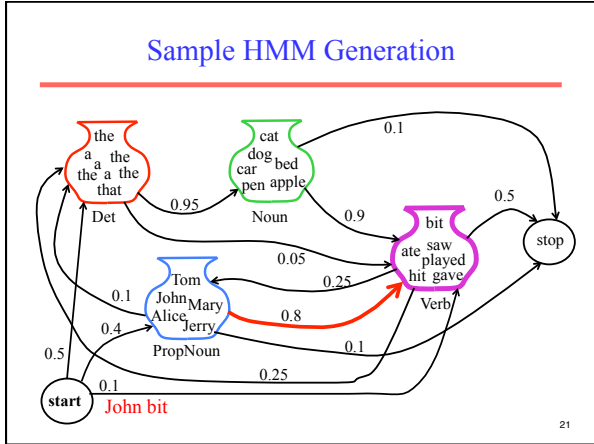
15

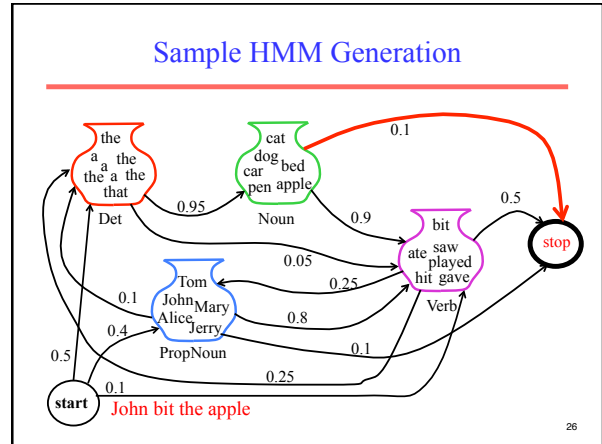
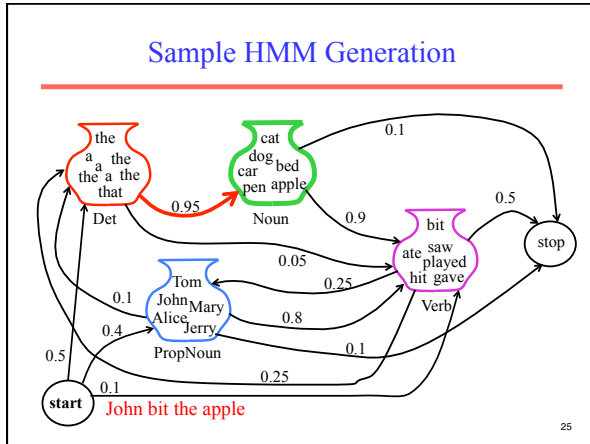
Sample HMM for POS



16







Formal Definition of an HMM

- A set of $N + 2$ states $S = \{s_0, s_1, s_2, \dots, s_N, s_F\}$
 - Distinguished start state: s_0
 - Distinguished final state: s_F
- A set of M possible observations $V = \{v_1, v_2, \dots, v_M\}$
- A state transition probability distribution $A = \{a_{ij}\}$

$$a_{ij} = P(q_{t+1} = s_j | q_t = s_i) \quad 1 \leq i, j \leq N \text{ and } i = 0, j = F$$

$$\sum_{j=1}^N a_{ij} + a_{iF} = 1 \quad 0 \leq i \leq N$$
- Observation probability distribution for each state j

$$B = \{b_j(k)\}$$

$$b_j(k) = P(v_k \text{ at } t | q_t = s_j) \quad 1 \leq j \leq N \quad 1 \leq k \leq M$$
- Total parameter set $\lambda = \{A, B\}$

27

HMM Generation Procedure

- To generate a sequence of T observations:

$$O = o_1 o_2 \dots o_T$$

Set initial state $q_1 = s_0$
 For $t = 1$ to T
 Transit to another state $q_{t+1} = s_j$ based on transition distribution a_{ij} for state q_t
 Pick an observation $o_t = v_k$ based on being in state q_t using distribution $b_{q_t}(k)$

28

Three Useful HMM Tasks

- **Observation Likelihood:** To classify and order sequences.
- **Most likely state sequence (Decoding):** To tag each token in a sequence with a label.
- **Maximum likelihood training (Learning):** To train models to fit empirical training data.

29

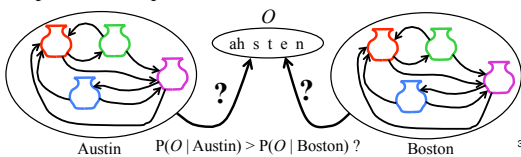
HMM: Observation Likelihood

- Given a sequence of observations, O , and a model with a set of parameters, λ , what is the probability that this observation was generated by this model: $P(O|\lambda)$?
- Allows HMM to be used as a **language model**: A formal probabilistic model of a language that assigns a probability to each string saying how likely that string was to have been generated by the language.
- Useful for two tasks:
 - Sequence Classification
 - Most Likely Sequence

30

Sequence Classification

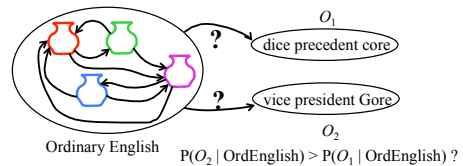
- Assume an HMM is available for each category (i.e. language).
- What is the most likely category for a given observation sequence, i.e. which category's HMM is most likely to have generated it?
- Used in speech recognition to find most likely word model to have generate a given sound or phoneme sequence.



31

Most Likely Sequence

- Of two or more possible sequences, which one was most likely generated by a given model?
- Used to score alternative word sequence interpretations in speech recognition.



32

HMM: Observation Likelihood Naïve Solution

- Consider all possible state sequences, Q , of length T that the model could have traversed in generating the given observation sequence.
- Compute the probability of a given state sequence from A , and multiply it by the probabilities of generating each of given observations in each of the corresponding states in this sequence to get $P(O, Q | \lambda) = P(O | Q, \lambda) P(Q | \lambda)$.
- Sum this over all possible state sequences to get $P(O | \lambda)$.
- Computationally complex: $O(TN^T)$.

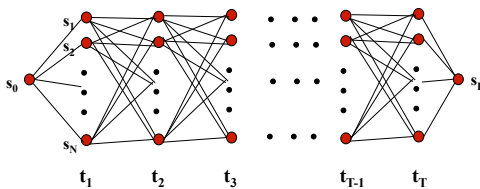
33

HMM: Observation Likelihood Efficient Solution

- Due to the Markov assumption, the probability of being in any state at any given time t only relies on the probability of being in each of the possible states at time $t-1$.
- **Forward Algorithm:** Uses dynamic programming to exploit this fact to efficiently compute observation likelihood in $O(TN^2)$ time.
 - Compute a *forward trellis* that compactly and implicitly encodes information about all possible state paths.

34

Forward Trellis



- Continue forward in time until reaching final time point and sum probability of ending in final state.

35

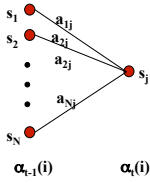
Forward Probabilities

- Let $\alpha_t(j)$ be the probability of being in state j after seeing the first t observations (by summing over all initial paths leading to j).

$$\alpha_t(j) = P(o_1, o_2, \dots, o_t, q_t = s_j | \lambda)$$

36

Forward Step



- Consider all possible ways of getting to s_j at time t by coming from all possible states s_i and determine probability of each.
- Sum these to get the total probability of being in state s_j at time t while accounting for the first $t-1$ observations.
- Then multiply by the probability of actually observing o_t in s_j .

37

Computing the Forward Probabilities

- Initialization

$$\alpha_1(j) = a_{0j} b_j(o_1) \quad 1 \leq j \leq N$$

- Recursion

$$\alpha_t(j) = \left[\sum_{i=1}^N \alpha_{t-1}(i) a_{ij} \right] b_j(o_t) \quad 1 \leq j \leq N, \quad 1 < t \leq T$$

- Termination

$$P(O | \lambda) = \alpha_{T+1}(s_F) = \sum_{i=1}^N \alpha_T(i) a_{iF}$$

38

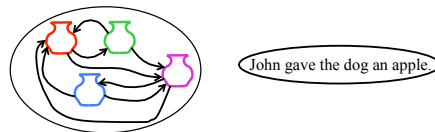
Forward Computational Complexity

- Requires only $O(TN^2)$ time to compute the probability of an observed sequence given a model.
- Exploits the fact that all state sequences must merge into one of the N possible states at any point in time and the Markov assumption that only the last state effects the next one.

39

Most Likely State Sequence (Decoding)

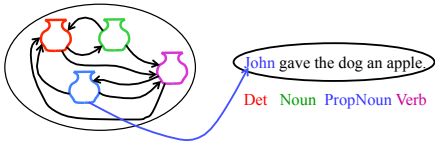
- Given an observation sequence, O , and a model, λ , what is the most likely state sequence, $Q=q_1, q_2, \dots, q_T$, that generated this sequence from this model?
- Used for sequence labeling, assuming each state corresponds to a tag, it determines the globally best assignment of tags to all tokens in a sequence using a principled approach grounded in probability theory.



40

Most Likely State Sequence

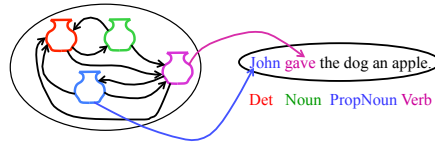
- Given an observation sequence, O , and a model, λ , what is the most likely state sequence, $Q=q_1, q_2, \dots, q_T$, that generated this sequence from this model?
- Used for sequence labeling, assuming each state corresponds to a tag, it determines the globally best assignment of tags to all tokens in a sequence using a principled approach grounded in probability theory.



41

Most Likely State Sequence

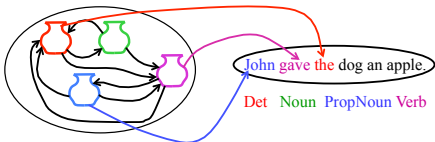
- Given an observation sequence, O , and a model, λ , what is the most likely state sequence, $Q=q_1, q_2, \dots, q_T$, that generated this sequence from this model?
- Used for sequence labeling, assuming each state corresponds to a tag, it determines the globally best assignment of tags to all tokens in a sequence using a principled approach grounded in probability theory.



42

Most Likely State Sequence

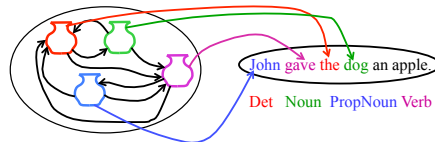
- Given an observation sequence, O , and a model, λ , what is the most likely state sequence, $Q=q_1, q_2, \dots, q_T$, that generated this sequence from this model?
- Used for sequence labeling, assuming each state corresponds to a tag, it determines the globally best assignment of tags to all tokens in a sequence using a principled approach grounded in probability theory.



43

Most Likely State Sequence

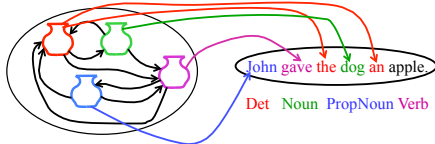
- Given an observation sequence, O , and a model, λ , what is the most likely state sequence, $Q=q_1, q_2, \dots, q_T$, that generated this sequence from this model?
- Used for sequence labeling, assuming each state corresponds to a tag, it determines the globally best assignment of tags to all tokens in a sequence using a principled approach grounded in probability theory.



44

Most Likely State Sequence

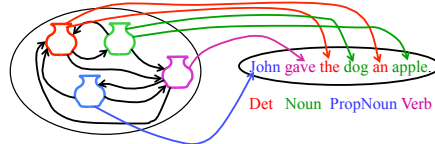
- Given an observation sequence, O , and a model, λ , what is the most likely state sequence, $Q=q_1, q_2, \dots, q_T$, that generated this sequence from this model?
- Used for sequence labeling, assuming each state corresponds to a tag, it determines the globally best assignment of tags to all tokens in a sequence using a principled approach grounded in probability theory.



45

Most Likely State Sequence

- Given an observation sequence, O , and a model, λ , what is the most likely state sequence, $Q=q_1, q_2, \dots, q_T$, that generated this sequence from this model?
- Used for sequence labeling, assuming each state corresponds to a tag, it determines the globally best assignment of tags to all tokens in a sequence using a principled approach grounded in probability theory.



46

HMM: Most Likely State Sequence Efficient Solution

- Obviously, could use naïve algorithm based on examining every possible state sequence of length T .
- Dynamic Programming can also be used to exploit the Markov assumption and efficiently determine the most likely state sequence for a given observation and model.
- Standard procedure is called the **Viterbi algorithm** (Viterbi, 1967) and also has $O(N^2T)$ time complexity.

47

Viterbi Scores

- Recursively compute the probability of the most likely subsequence of states that accounts for the first t observations and ends in state s_j .

$$v_t(j) = \max_{q_0, q_1, \dots, q_{t-1}} P(q_0, q_1, \dots, q_{t-1}, o_1, \dots, o_t, q_t = s_j | \lambda)$$

- Also record “backpointers” that subsequently allow backtracing the most probable state sequence.
 - $b_t(j)$ stores the state at time $t-1$ that maximizes the probability that system was in state s_j at time t (given the observed sequence).

48

Computing the Viterbi Scores

- Initialization

$$v_1(j) = a_{0j} b_j(o_1) \quad 1 \leq j \leq N$$

- Recursion

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

- Termination

$$P^* = v_{T+1}(s_F) = \max_{i=1}^N v_T(i) a_{iF}$$

Analogous to Forward algorithm except take *max* instead of *sum*

49

Computing the Viterbi Backpointers

- Initialization

$$bt_1(j) = s_0 \quad 1 \leq j \leq N$$

- Recursion

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

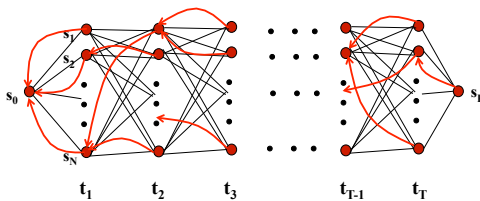
- Termination

$$q_T^* = bt_{T+1}(s_F) = \operatorname{argmax}_{i=1}^N v_T(i) a_{iF}$$

Final state in the most probable state sequence. Follow backpointers to initial state to construct full sequence.

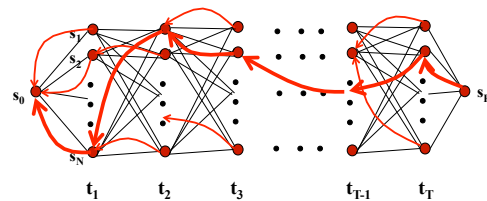
50

Viterbi Backpointers



51

Viterbi Backtrace



Most likely Sequence: $s_0 s_N s_1 s_2 \dots s_2 s_F$

52

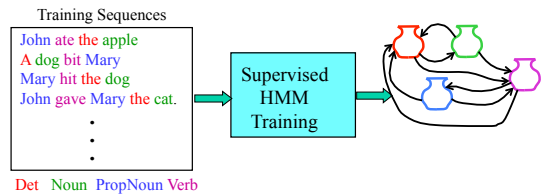
HMM Learning

- **Supervised Learning:** All training sequences are completely labeled (tagged).
- **Unsupervised Learning:** All training sequences are unlabelled (but generally know the number of tags, i.e. states).
- **Semisupervised Learning:** Some training sequences are labeled, most are unlabeled.

53

Supervised HMM Training

- If training sequences are labeled (tagged) with the underlying state sequences that generated them, then the parameters, $\lambda = \{A, B\}$ can all be estimated directly.



54

Supervised Parameter Estimation

- Estimate state transition probabilities based on tag bigram and unigram statistics in the labeled data.

$$a_{ij} = \frac{C(q_i = s_i, q_{i+1} = s_j)}{C(q_i = s_i)}$$

- Estimate the observation probabilities based on tag/word co-occurrence statistics in the labeled data.

$$b_j(k) = \frac{C(q_i = s_j, o_i = v_k)}{C(q_i = s_j)}$$

- Use appropriate smoothing if training data is sparse.

55

Learning and Using HMM Taggers

- Use a corpus of labeled sequence data to easily construct an HMM using supervised training.
- Given a novel unlabeled test sequence to tag, use the Viterbi algorithm to predict the most likely (globally optimal) tag sequence.

56

Evaluating Taggers

- Train on *training set* of labeled sequences.
- Possibly tune parameters based on performance on a *development set*.
- Measure accuracy on a disjoint *test set*.
- Generally measure *tagging accuracy*, i.e. the percentage of tokens tagged correctly.
- Accuracy of most modern POS taggers, including HMMs is 96–97% (for Penn tagset trained on about 800K words).
 - Generally matching human agreement level.

57