Natural Language Processing

Part 2: Part of Speech Tagging

Word classes- 1

- Words can be grouped into classes referred to as Part of Speech (PoS) or morphological classes
  - Traditional grammar is based on few types of PoS (noun, verb, adjective, preposition, adverb, conjunction, etc..)
  - More recent models are based on a larger number of classes
    - 45 Penn Treebank
    - 87 Brown corpus
    - 146 C7 tagset
  - The word PoS provides crucial information to determine the roles of the word itself and of the words close to it in the sentence
    - knowing if a word is a personal pronoun (*I*, *you*, *he/she*,..) or a possessive pronoun (*my*, *your*, *his/her*,...) allows a more accurate selection of the most probable words that appear in its neighborhood (the syntactic rules are often based on the PoS of words)
    - e.g. *possessive pronoun - noun* vs. *personal pronoun - verb*

Word classes - 2

- Classes are usually defined on the basis of the morphological properties or of the syntactic role of words
  - words that have similar use given their affixes (morphological properties)
  - words that share similar contexts (properties related to their statistical distribution/syntactical role)
- Usually the classes are not defined on the property of semantic coherence
  - a noun is a referrer for “persons, places or things”
- The considered classes can be closed or open
  - Closed classes are those containing a fixed set of items (es. prepositions)
    - The usually contain function words (of, and, that, from, in, by,…) that are short, frequent and have a specific role in the grammar
  - Open classes are instead prone to the addition of new terms (e.g. verbs and nouns)

Word classes- nouns/names

- The 4 largest open classes of words, present in most of the languages, are
  - nouns
  - verbs
  - adverbs
  - adjectives
- Nouns are concrete terms (e.g. ship, table), abstractions (e.g. relationship, function), verb-like terms (e.g. pacing, pricing)
  - They can be functionally tied to determiners (the ship, a ship, ..) and they can assume the plural form (the ships), etc..
  - They are traditionally divided into proper nouns (e.g. Marco, Italy) and common nouns (e.g. book, lecture)
  - In many languages common nouns are also divided count nouns (they have the plural form) and mass nouns (they are used only in their singular form, e.g. snow, communism)

Word classes- verbs & adjectives

- The class of verbs includes most of the words that refer to actions and processes
  - to write, to go, to eat
  - they have “some” morphological inflections
    - In English non-3rd-person-sg (eat), 3rd-person-sg (eats), progressive (eating), past-participle (eaten), past perfect (ate)
    - A special class of verbs is that of auxiliary verbs (to be, to have)
- The class of adjectives contains terms describing properties or qualities
  - Most of the languages have adjectives for concepts like color (white, red,..), age (young, old,..) quality (good, bad,...), etc..

Word classes- adverbs

- Usually adverbs are used to modify other terms (not only verbs)
  - Directional or locative adverbs specify the direction or location of a given action (here, there, up, ..)
  - Degree adverbs specify the extent of an action, process or property (extremely, very,...)
  - Manner adverbs describe the modality of some action or process (slowly, delicately, smartly,..)
  - Temporal adverbs describe the time for an action or event (yesterday, today, before, after, later, Monday,...)
- The class of adverbs is somehow heterogeneous
  - Some adverbs are similar to nouns (e.g. Monday – we will meet Monday, we meet on Mondays)

Word classes- closed classes 1

- The closed classes are the most different among languages
  - Prepositions: from, to, on, of, with, for, by, at, …
  - Determiners: the, a , an (il, la, lo, le, i, gli, un,..)
  - Pronouns: he, she, I, who, others,…
  - Conjunctions: and, but, or, if, because, when,…
  - Auxiliary verbs: be, have, can, must,…
  - Numerals: one, two,.., first, second
  - Particles: up, down, on, off, in, out, at, by (e.g. turn off)
- Prepositions occur before noun phrases
  - Semantically they express a relationship (spatial, temporal, etc..)
  - In English some prepositions assume a different role in predefined contexts and they are considered in the special class of particles
    - e.g. *on* in verbal phrases as "*go on*" where they have a role like an adverb

Word classes- closed classes 2

- The determiners are often at the beginning of a noun phrase
  - They are among the most common terms (e.g. the in English)
- Conjunctions are used to connect phrases, clauses or sentences
  - The coordinating conjunctions are used to join two elements at the same level (for, and, nor, but, or, yet, so are the 6 most frequent)
    - copulative (and ,also,..), disjunctive (or, nor, ..), adversative (but, however, still, yet..), illative (for, so,...), correlative (both...and, either...or, neither...nor,..)
  - Subordinating conjunctions are sued to express a fact that depends on a main clause (they define a relation between two clauses)
    - condition (unless, provided that, if, even if), reason (because, as, as if), choice (rather than, than, whether), contrast (though, although, even though, but), location (where, wherever), result/effect (in order that, so, so that, that), time (while, once, when, since, whenever, after, before, until, as soon as), concession and comparison (although, as, as though, even though, just as, though, whereas, while)

Word classes- closed classes 3

- Pronouns are short elements that are used to refer noun phrases, entities or events
  - Personal pronouns refer to persons or entities (I, you, me,..)
  - Possessive pronouns define the possess or, in general, an abstract relation between a person and an (abstract) object (my, his/her, your,..)
  - Relative pronouns are used to relate two sentences by subordinating the sentence they start with respect to the sentence containing the referred word (who, whom,...)
  - Demonstrative pronouns refer to a person or object given a spatial or temporal relation (this, that, these, those,..)
  - Indefinite pronouns are used to refer a generic object, person, event (none, nobody, everybody, someone, each one....)

Word classes- closed classes 4

- Auxiliary verbs are used in combination with other verbs to give a particular meaning to the verbal phrase (have, be, do will)
  - they are used to define the compound verb tenses (present perfect, future perfect, ..)
    - he has eaten an apple, he will go home
  - they are used to form a question or a negative form of a verb
    - *I do not (don't) walk, Do you like it?*
  - "be" is used to define the passive voice of verbs (the apple is eaten)
  - they can express a modality for the action (modal auxiliary)
    - need/requirement (must, have to, need to)
    - possibility (may)
    - will (will, wish)
    - capacity (can)

Tagsets

- Some different tag sets have been proposed for PoS tagging
  - The tagset for English have a different detail level
    - Penn Treebank tagset: 45 tags (Marcus et al. 1993)
    - C5 tagset: 61 tags (CLAWS project by Lacaster UCREL, 1997)
    - C7 tagset: 146 tags (Leech et al. 1994)
  - Tags are usually specified at the word end after /

The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

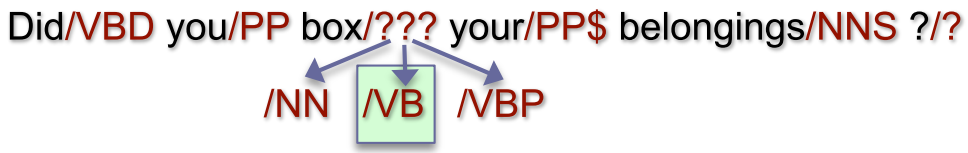
  - The Penn Treebank tagset does not describe some properties that can be derived from the analysis of the lexical entity or from syntax
    - e.g. prepositions and subordinating conjunctions are combined into the same tag IN since the are disambiguated in the syntactical parse tree

Penn Treebank tagset

Tag	Description	Example	Tag	Description	Example
CC	Coordin. Conjunction	*and, but, or*	SYM	Symbol	*+, %, €*
CD	Cardinal number	*one, two*	TO	"to"	*to*
DT	Determiner	*a, the*	UH	Interjection	*ah, uh, oops*
EX	Existential 'there"	*there*	VB	Verb, base form	*eat*
FW	Foreign word	*mea culpa*	VBD	Verb, past tense	*ate*
IN	Preposition/sub-conj	*of, in, by*	VBG	Verb, gerund	*eating*
JJ	Adjective	*yellow*	VBN	Verb, past particip.	*eaten*
JJR	Adj. comparative	*bigger*	VBP	Verb, non-3sg pres	*eat*
JJS	Adj. superlative	*biggest*	VBZ	Verb, 3sg pres	*eats*
LS	List item marker	*1,2,3*	WDT	Wh-determiner	*which, that*
MD	Modal	*can, should*	WP	Wh-pronoun	*what, who*
NN	Noun, singular/mass	*dog, snow*	WP$	Possessive wh-	*whose*
NNS	Noun, plural	*dogs*	WRB	Wh-adverb	*how, where*
NNP	Proper noun, singul.	*Marco*	$	Dollar sign	*$*
NNPS	Proper noun, plural	*Alps*	#	Pound sign	#
PDT	Predeterminer	*all, both*	"	Left quote	"
POS	Possessive ending	*'s*	"	Right quote	"
PP	Personal pronoun	*I, you, he*	(	Left parenthesis	*([{ <*
PP$	Possessive pronoun	*my, your*	)	Right parenthesis	*)] } >*
RB	Adverb	*never, often*	,	Comma	*,*
RBR	Adverb, comparative	*faster*	.	Sentence-final pun	*. ! ?*
RBS	Adverb, superlative	*fastest*	:	Mid-sentence punt.	*: ; … -*
RP	Particle	*up, on ,off*			

PoS tagging & tags

- PoS tagging consists in assigning a tag to each word in a document
  - The selection of the employed tagset depends on the language and specific application
  - The input is a word sequence and the employed tagset while the output is the association of each word to its "*best*" tag
  - There may exist more tags for a given word (ambiguity)



  - The PoS tagger task is to solve these ambiguities by selecting the most appropriate tag given the word context
    - The percentage of ambiguous words is not too high, but among them there are very frequent words (e.g. *can* – Auxiliary verb, Noun, Verb, *still* has 7 compatible tags – adj, adv, verb, noun)

PoS tagging algorithms

- Rule-based taggers
  - The are based on the "handcrafting" of a large rule-base that specifies the conditions to be verified to assign a specific tag in the ambiguous cases
    - e.g. a word is a noun if it is preceded by a determiner
- Probabilistic taggers
  - They revolve ambiguities by estimating the probability that a given word as a specific tag in the observed context. The parameters for the probability model are estimated on a reference corpus.
- Other approaches
  - tagging can be cast as a classification task (each tag corresponds to a class and the classifiers exploits features that describe the context – e.g. features of the words on the left/right of the considered word)
  - Taggers can exploits rules learnt from examples

Rule-based PoS tagging

- Two step process (e.g. ENGTWOL Voutilainen, 1995)
  - Word tagging using a lexicon (more than one tag can be assigned to each word) exploiting morphological/orthographic rules

text	stem	PoS	PoS features
Pavlov	**PAVLOV**	**N**	**NOM SG PROPER**
had	**HAVE** HAVE	**V** PCP2	**PAST VFIN SVO** SVO
shown	**SHOW**	**PCP2**	**SVOO SVO SV**
that	THAT THAT THAT **THAT**	ADV PRON DET **CS**	 DEM SG CENTRAL DEM SG
salivation	**SALIVATION**	**N**	**NOM SG**

N: Noun
V: Verb
PCP2: Past Participle
ADV: Adverb
PRON: Pronoun
DET: Determiner
CS: Subordinating Conjunction

SVOO: Subject-Verb-Object-Object
NOM: non-genitive
VFIN: finite verb
DEM: demonstrative

  - Application of rules to select only one tag among those assigned to each word (rules exploit the word context)

Rules

- Rules are aimed to removing the cases that are not compatible with the context
  - In ENGTWOL there are about 1100 rules

```
ADVERBIAL-THAT RULE
input: "that"
if
  (+1 A/ADV/QUANT); // the following word is an adjective, adverb orquantifier
  (+2 SENT-LIM);    // and the following one is a sentence boundary
  (NOT -1 SVOC/A);  // and the preceding word is a verb like "consider"
                    // that admits an adjective as object (I consider that good)
then eliminate non-ADV tags
else eliminate ADV tag
```

  - ENGTWOL has also probabilistic constraints and it may exploit syntactic information

Probabilistic tagging with HMM

- Given a word sequence an HMM-based tagger computes the tag sequence maximizing its probability
  - The probability is assigned to the whole tag sequence T
    - it is the tag sequence yielding the maximum likelihood given the observed word sequence W (Viterbi)

$$\hat{T} = \text{argmax}_T p(T|W) \quad T = t_1 t_2 \ldots t_n \quad W = w_1 w_2 \ldots w_n$$

  - By the Bayes rule the previous expression can be rewritten as

$$\hat{T} = \text{argmax}_T \frac{p(T)p(W|T)}{P(W)} = \text{argmax}_T p(T)p(W|T)$$

HMM tagging – model assumptions

- From the chain rule for probability factorization

$$p(T)p(W|T) = \prod_{i=1}^{n} p(w_i|w_1t_1 \ldots w_{i-1}t_{i-1}t_i)p(t_i|w_1t_1 \ldots w_{i-1}t_{i-1})$$

- Some approximation are introduced to simplify the model, such as
  - The word probability depends only on the tag

$$p(w_i|w_1t_1 \ldots w_{i-1}t_{i-1}t_i) = p(w_i|t_i)$$

  - The dependence of a tag from the preceding tag history is limited in time, f.i. a tag depends only on the two preceding ones

$$p(t_i|w_1t_1 \ldots w_{i-1}t_{i-1}) = p(t_i|t_{i-2}t_{i-1})$$

HMM tagging – model and parameter estimation

- With the considered assumption the optimal tag sequence maximizes

$$p(t_1)p(t_2|t_1)\prod_{i=3}^{n} p(t_i|t_{i-2}t_{i-1})\prod_{i=1}^{n} p(w_i|t_i)$$

  - The required probabilities can be estimated by counting occurrences on a labeled dataset with adequate smoothing/backoff techniques

$$p(t_i|t_{i-2}t_{i-1}) \approx \frac{\#(t_{i-2}t_{i-1}t_i)}{\#(t_{i-2}t_{i-1})} \quad p(w_i|t_i) \approx \frac{\#(w_i t_i)}{\#t_i}$$

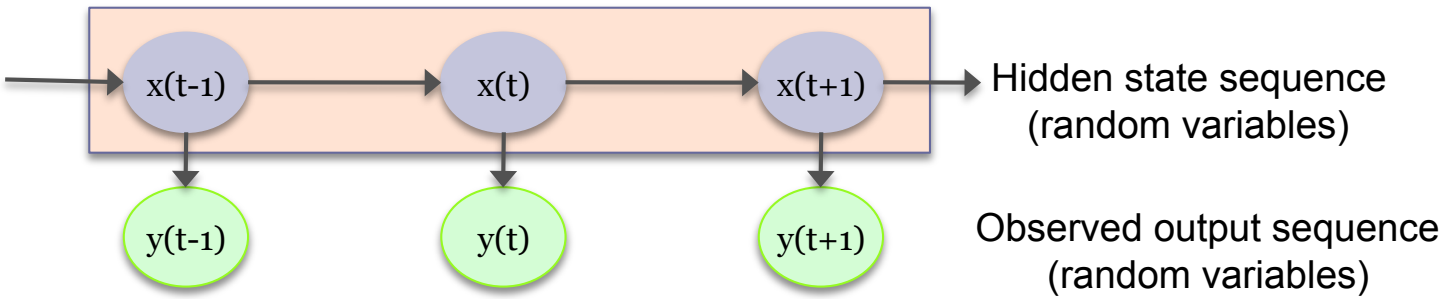
  - The proposed model is an HMM of order 2 whose states correspond to the tags and the observations to words
  - The optimal state sequence (tag) can be computed with the Viterbi algorithm
  - This approach yields and accuracy of about 96% (Weischedel et al. 1993; DeRose 1988)

Unknown words

- The PoS tagging algorithms exploit a dictionary that lists all the tags that can be assigned to each word
- In presence of a unknown word (name, acronym, new word)
  - The tagger may exploit the context tags to select the most probable tag
  - It can be assumed that all the tags can be selected with equal probability
  - Otherwise the tag distribution for rare words in the training corpus can be used (for instance those occurring only once)
    - The most probable tag is noun, than verb
  - Also morphological information can be exploited
    - English words ending in –s are likely to be plural nouns
    - words beginning with a capital letter are likely to be proper nouns
    - some word classes have standard suffixes that my provide an hint (-ion –al –ive –ly)

Hidden Markov Models

- An HMM is a probabilistic model of a system characterized by a finite set of non-observable states
  - The observable event is the output that depends on the state
  - Each state is characterized by a specific probability distribution for the output values
  - The observable output sequence carries information on the state trajectory of the system to generate it, but the state sequence is "hidden"
  - The state evolution is modeled by a Markovian process



HMM - definition

- A Hidden Markov Model (of order 1) is defined by
  - A finite set of N states $Q=\{q_1,q_2,\ldots,q_N\}$
  - A set of transition probabilities organized into a transition matrix $A=\{a_{ij}\}_{i,j=1,..,N}$ being

$$a_{ij} = p(x(t) = q_i | x(t-1) = q_j)$$

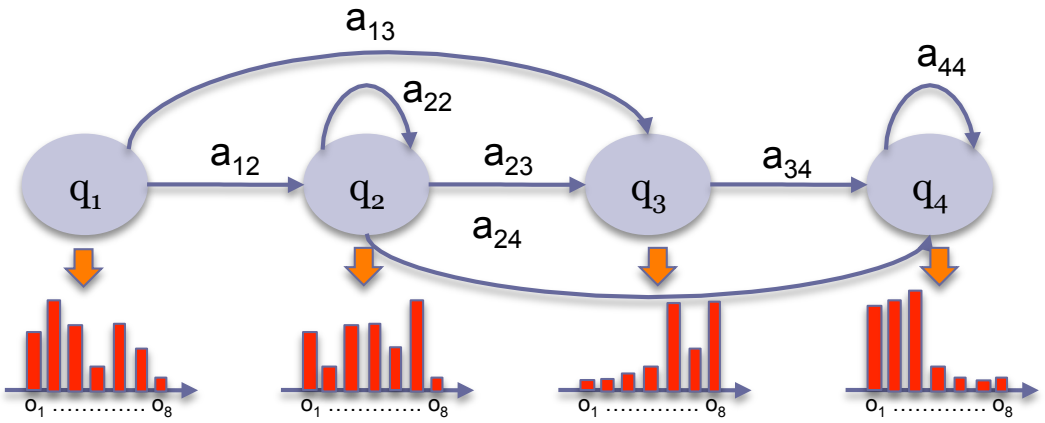
  - An initial probability distribution π on Q, such that

$$\pi_i = p(x(0) = q_i)$$

  - A set of output distributions $B = \{b_i(o_k)\}_{i=1,..,N\ k=1,\ldots,m}$ that define the probabilities of emitting a given symbol o_k when the model is in state q_i

$$b_i(o_k) = p(o_k | q_i)$$

HMM - tasks



- Given an observed sequence $O=o_1o_2....o_T$ estimate the probability that is was generated by the model
- Given an observed sequence $O=o_1o_2....o_T$ estimate the state sequence $x_1x_2....x_T$ that generate it given the model
- Given a set of sequences generated by the model, $O_k=o_{1k}o_{2k}....o_{Tkk}$, estimate the model parameters A, B, π in order to maximize the likelihood of all the sequences O_k given the trained model

HMM - p(O|M)

- Computation of the probability of a given observed sequence for a given model
  - It can be formulated with an efficient scheme (forward algorithm)
  - The α coefficients are defined as the joint probability of the observed partial sequence $o_1^t = o_1 o_2 \ldots o_t$ and the state q_i at time t

$$\alpha_t(o_1^t, q_i) = p(o_1 \ldots o_t, x(t) = q_i | M)$$

  - The α coefficient can be iteratively computed starting from the initial distribution π
    - at t=1

$$\alpha_1(o_1, q_i) = \pi_i \cdot b_i(o_1)$$

probability of state $x(0)=q_i$

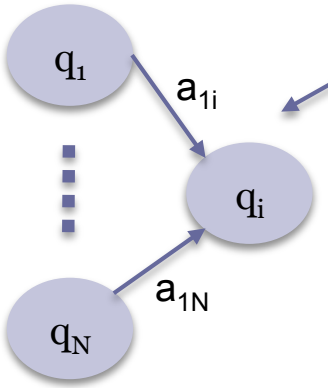
probability of generating o_1 in state $x(0)=q_i$

HMM – forward algorithm1

- Iterative step

$$\alpha_{t+1}(o_1^{t+1}, q_i) = \left[\sum_{j=1}^{N} a_{ji}\alpha_t(o_1^t, q_j)\right] \cdot b_i(o_{t+1})$$

probability of generating o_{t+1} in state $x(0)=q_i$



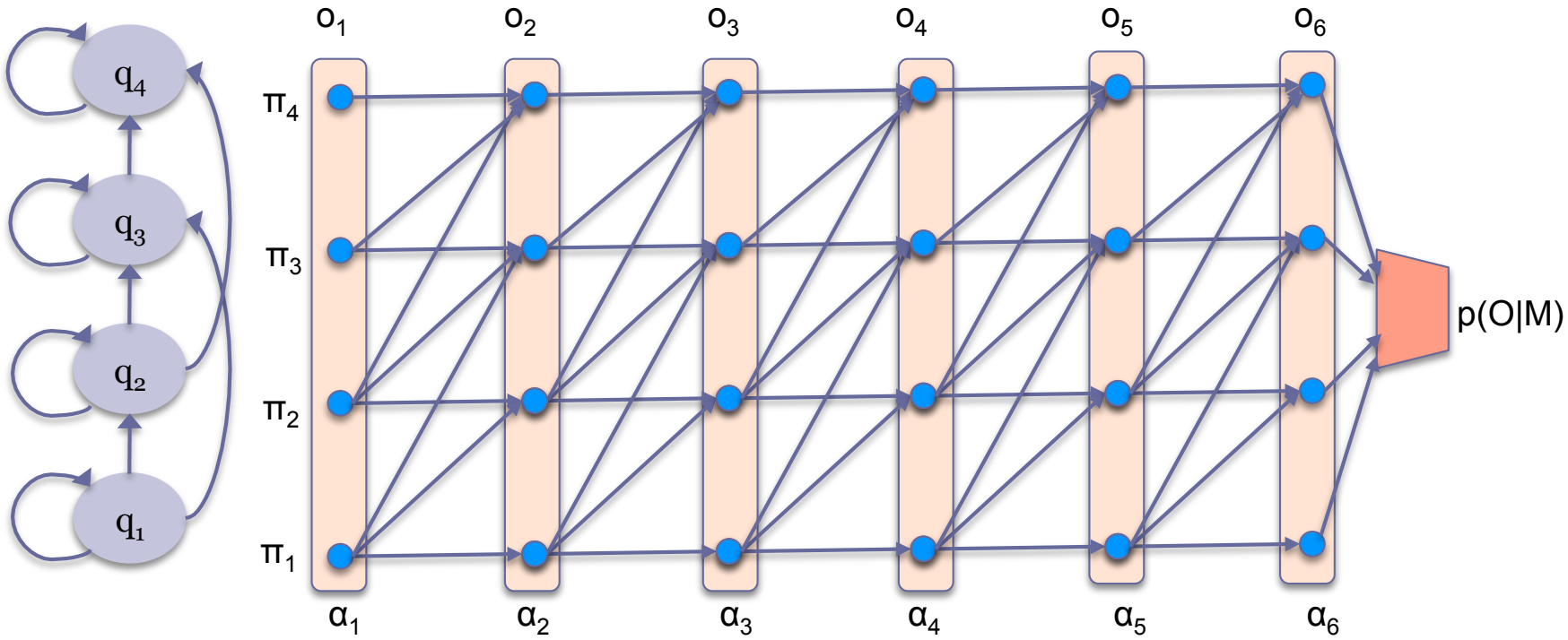
$$p(x(t+1) = q_i | x(t) = q_j) p(o_1^t, x(t) = q_j)$$

- Termination

$$p(o_1^T | M) = \sum_{i=1}^{N} \alpha_T(o_1^T, q_i)$$

HMM- forward algorithm 2

- The algorithm can be visualized with the forward graph
  - The complexity is of order $O(TN^2)$



HMM - alignment

- Which is the state sequence that better explains an observation
  - It yields the most likely alignment between the observation sequence O and a state sequence X having the same length
  - The Viterbi algorithm yields a solution maximizing P(X|O,M)
  - We define the followin variablesg

$$\delta_t(o_1^t, q_i) = \max_{x(1)\ldots x(t-1)} p(o_1 \ldots o_t, x(1) \ldots x(t-1), x(t) = q_i | M),$$

  where x(1)...x(t-1) is the most likely state sequence given the observations $o_1 \ldots o_t$ and the final state q_i
  - The optimal state sequence is stored in the variables

$$\psi_t(o_1^t, q_i) = \text{argmax}_{x(1)\ldots x(t-1)} p(o_1 \ldots o_t, x(1) \ldots x(t-1), x(t) = q_i | M),$$

HMM –Viterbi algorithm

- The procedure consists in the following steps
  - initialization

$$\delta_1(o_1^1, q_i) = \pi_i \cdot b_i(o_1) \quad \psi_1(o_1^1, q_i) = \emptyset$$

  - recursion

$$\delta_t(o_1^t, q_i) = \max_{1 \le j \le N} \left[\delta_{t-1}(o_1^{t-1}, q_j) a_{ji}\right] b_i(o_t)$$

$$\psi_t(o_1^1, q_i) = \text{argmax}_{1 \le j \le N} \left[\delta_{t-1}(o_1^{t-1}, q_j) a_{ji}\right]$$

  - termination

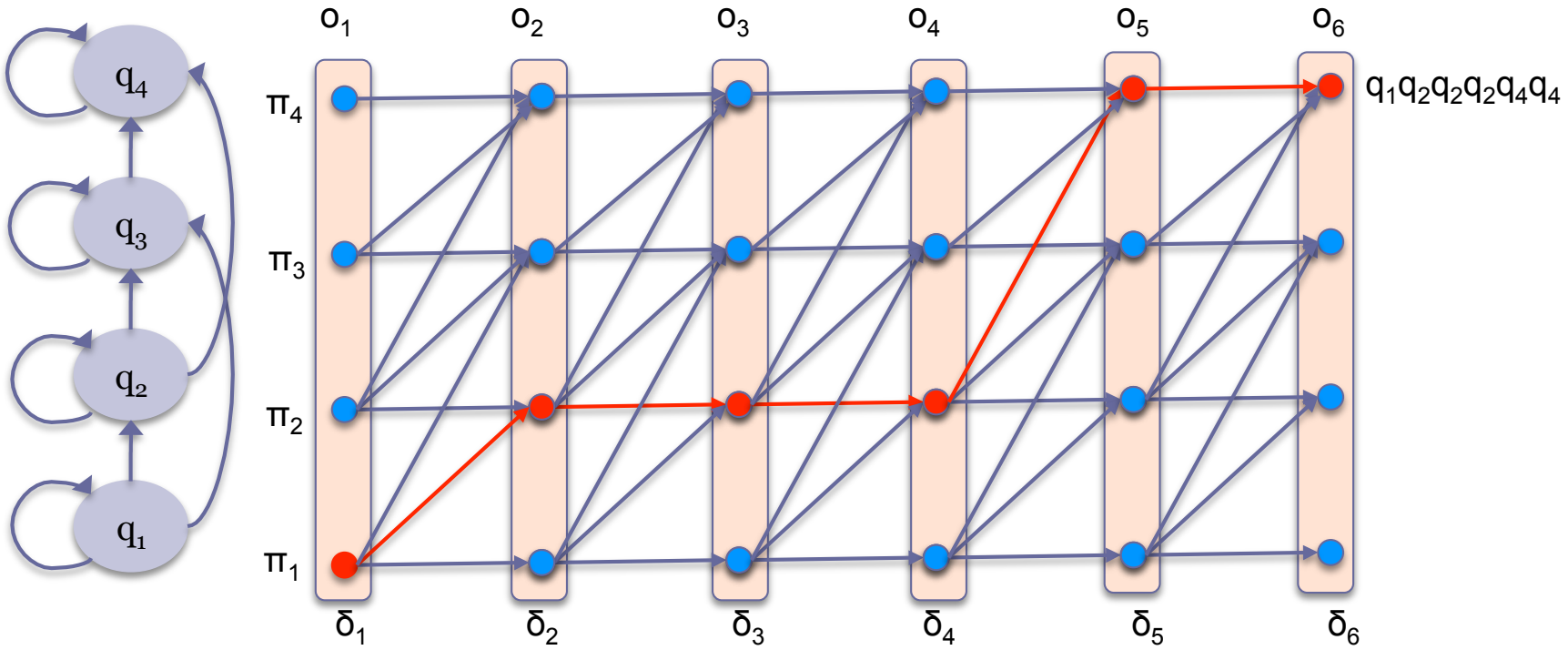
$$p^* = \max_{1 \le j \le N} \delta_T(o_1^T, q_j) \quad x^*(T) = \text{argmax}_{1 \le j \le N} \delta_T(o_1^T, q_j)$$

  - backtracking for computing the state sequence

$$x^*(t) = \psi_{t+1}(o_1^{t+1}, x^*(t+1))$$

HMM- Viterbi algorithm 2

- The algorithm determines the optimal path for states on the transition trellis



HMM - training

- The model parameters $M=(A,B,\pi)$ can be estimated on a set of given sequences
  - If we assume that the available sequences are independent on each other, we can maximize the probability of this set of sequences given the model
  - The is no closed form solution to this problem and the available algorithms approximate iteratively the solution (without any guarantee that the global optimum is obtained)
  - The Baum-Welch algorithm exploits the EM (Expectation Maximization) procedure
  - The problem difficulty depends on the fact that the state sequence is not known but it must be estimated. Hence, the classical methods for Maximum Likelihood estimation, based on the approximation of the probabilities with the observed frequencies, can not be used
    - How can we obtain the frequency of the transitions from i to j to estimate a_{ij}?