CMPE 297, INSTRUCTOR: JORJETA JETCHEVA

SEQUENCE LABELING FOR POS AND NAMED ENTITIES

INTRODUCTION

- PoS Tagging involves assigning part of speech tags to words in a sequence (e.g., NOUN, VERB)
  - Applications of POS include speech synthesis (e.g., **record** is pronounced differently when it is a verb vs. noun), machine translation (order of words is affected by what part of speech they are), etc.
- NER (Named Entity Recognition) involves assigning tags to words or phrases such as PERSON, LOCATION, or ORGANIZATION
  - Knowing if a named entity like **Washington** is a name of a person, a place, or a university is important to many natural language processing tasks like question answering.
- Both PoS Tagging and NER are sequence labeling tasks: involve assigning a label to each word in a sequence, and the input and output sequences are thus of the same length
- Sequence labeling approaches we will cover:
  - Hidden Markov Model (HMM)
  - Conditional Random Fields (CRF)

PARTS OF SPEECH

- Example tagging systems

	Tag	Description	Example
Open Class	**ADJ**	Adjective: noun modifiers describing properties	*red, young, awesome*
	ADV	Adverb: verb modifiers of time, place, manner	*very, slowly, home, yesterday*
	NOUN	words for persons, places, things, etc.	*algorithm, cat, mango, beauty*
	VERB	words for actions and processes	*draw, provide, go*
	PROPN	Proper noun: name of a person, organization, place, etc..	*Regina, IBM, Colorado*
	INTJ	Interjection: exclamation, greeting, yes/no response, etc.	*oh, um, yes, hello*
Closed Class Words	**ADP**	Adposition (Preposition/Postposition): marks a noun's spacial, temporal, or other relation	*in, on, by, under*
	AUX	Auxiliary: helping verb marking tense, aspect, mood, etc.,	*can, may, should, are*
	CCONJ	Coordinating Conjunction: joins two phrases/clauses	*and, or, but*
	DET	Determiner: marks noun phrase properties	*a, an, the, this*
	NUM	Numeral	*one, two, first, second*
	PART	Particle: a preposition-like form used together with a verb	*up, down, on, off, in, out, at, by*
	PRON	Pronoun: a shorthand for referring to an entity or event	*she, who, I, others*
	SCONJ	Subordinating Conjunction: joins a main clause with a subordinate clause such as a sentential complement	*that, which*
Other	**PUNCT**	Punctuation	; , ()
	SYM	Symbols like $ or emoji	$, %
	X	Other	asdf, qwfg

Figure 8.1 The 17 parts of speech in the Universal Dependencies tagset (Nivre et al., 2016a). Features can be added to make finer-grained distinctions (with properties like number, case, definiteness, and so on).

Tag	Description	Example	Tag	Description	Example	Tag	Description	Example
CC	coord. conj.	*and, but, or*	NNP	proper noun, sing.	*IBM*	TO	"to"	*to*
CD	cardinal number	*one, two*	NNPS	proper noun, plu.	*Carolinas*	UH	interjection	*ah, oops*
DT	determiner	*a, the*	NNS	noun, plural	*llamas*	VB	verb base	*eat*
EX	existential 'there'	*there*	PDT	predeterminer	*all, both*	VBD	verb past tense	*ate*
FW	foreign word	*mea culpa*	POS	possessive ending	*'s*	VBG	verb gerund	*eating*
IN	preposition/ subordin-conj	*of, in, by*	PRP	personal pronoun	*I, you, he*	VBN	verb past participle	*eaten*
JJ	adjective	*yellow*	PRP$	possess. pronoun	*your, one's*	VBP	verb non-3sg-pr	*eat*
JJR	comparative adj	*bigger*	RB	adverb	*quickly*	VBZ	verb 3sg pres	*eats*
JJS	superlative adj	*wildest*	RBR	comparative adv	*faster*	WDT	wh-determ.	*which, that*
LS	list item marker	*1, 2, One*	RBS	superlatv. adv	*fastest*	WP	wh-pronoun	*what, who*
MD	modal	*can, should*	RP	particle	*up, off*	WP$	wh-possess.	*whose*
NN	sing or mass noun	*llama*	SYM	symbol	+,%, &	WRB	wh-adverb	*how, where*

Figure 8.2 Penn Treebank part-of-speech tags.

PART-OF-SPEECH TAGGING

- Part-of-speech (PoS) tagging assigns a sequence of tags, $y_1, y_2, \ldots, y_n$ to a sequence of words $w_1 \ldots w_n$ (each word has 1 associated tag)
- Tagging involves disambiguation: words can function as multiple different part-of-speech (ambiguity) in general though have a unique part-of-speech role in a particular context
  - E.g., book can be a verb (book that flight) or a noun (hand me that book)
- Ambiguous words account for ~15% of the vocabulary but 55-67% of tokens in running text are ambiguous
- Example: below are 6 different tags for the word back

earnings growth took a **back/JJ** seat
a small building in the **back/NN**
a clear majority of senators **back/VBP** the bill
Dave began to **back/VB** toward the door
enable the country to buy **back/RP** debt
I was twenty-one **back/RB** then

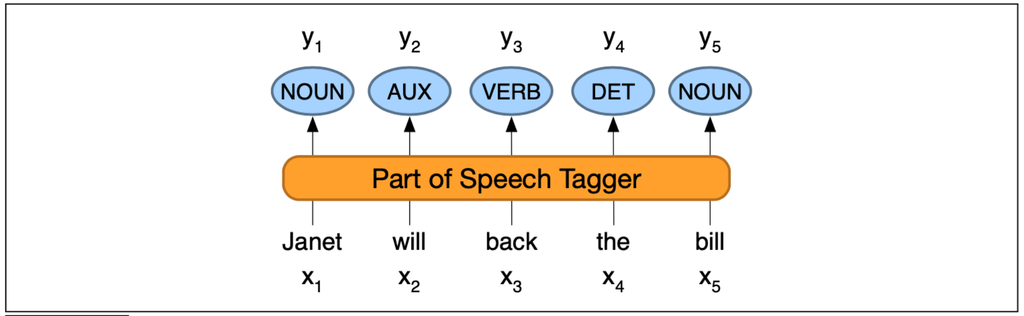

Figure 8.3 The task of part-of-speech tagging: mapping from input words $x_1, x_2, \ldots, x_n$ to output POS tags $y_1, y_2, \ldots, y_n$.

		WSJ		Brown	
Types:					
Unambiguous	(1 tag)	44,432	**(86%)**	45,799	**(85%)**
Ambiguous	(2+ tags)	7,025	**(14%)**	8,050	**(15%)**
Tokens:					
Unambiguous	(1 tag)	577,421	**(45%)**	384,349	**(33%)**
Ambiguous	(2+ tags)	711,780	**(55%)**	786,646	**(67%)**

Figure 8.4 Tag ambiguity in the Brown and WSJ corpora (Treebank-3 45-tag tagset).

Tag	Description	Example	Tag	Description	Example	Tag	Description	Example
CC	coord. conj.	*and, but, or*	NNP	proper noun, sing.	*IBM*	TO	"to"	*to*
CD	cardinal number	*one, two*	NNPS	proper noun, plu.	*Carolinas*	UH	interjection	*ah, oops*
DT	determiner	*a, the*	NNS	noun, plural	*llamas*	VB	verb base	*eat*
EX	existential 'there'	*there*	PDT	predeterminer	*all, both*	VBD	verb past tense	*ate*
FW	foreign word	*mea culpa*	POS	possessive ending	*'s*	VBG	verb gerund	*eating*
IN	preposition/ subordin-conj	*of, in, by*	PRP	personal pronoun	*I, you, he*	VBN	verb past partici-ple	*eaten*
JJ	adjective	*yellow*	PRP$	possess. pronoun	*your, one's*	VBP	verb non-3sg-pr	*eat*
JJR	comparative adj	*bigger*	RB	adverb	*quickly*	VBZ	verb 3sg pres	*eats*
JJS	superlative adj	*wildest*	RBR	comparative adv	*faster*	WDT	wh-determ.	*which, that*
LS	list item marker	*1, 2, One*	RBS	superlatv. adv	*fastest*	WP	wh-pronoun	*what, who*
MD	modal	*can, should*	RP	particle	*up, off*	WP$	wh-possess.	*whose*
NN	sing or mass noun	*llama*	SYM	symbol	*+,%, &*	WRB	wh-adverb	*how, where*

Figure 8.2 Penn Treebank part-of-speech tags.

PART-OF-SPEECH TAGGING (CONT.)

- Even for ambiguous words, different tags are not equally likely
- Most Frequent Class Baseline: given an ambiguous word, choose the tag that is most frequent for that word in the training corpus
- The accuracy of the Most Frequent Class Baseline is 92%!
- The state-of-the-art algorithms achieve 97%

NAMED ENTITIES & NAMED ENTITY TAGGING

- Most common named entity tags:
  - PER (person)
  - LOC (location)
  - ORG (organization)
  - GPE (geo-political entity)
- NER tags are also used for non-entities, e.g., dates, times, prices
- Domain-specific NER tags, e.g., for proteins, genes, products

Type	Tag	Sample Categories	Example sentences
People	PER	people, characters	**Turing** is a giant of computer science.
Organization	ORG	companies, sports teams	The **IPCC** warned about the cyclone.
Location	LOC	regions, mountains, seas	**Mt. Sanitas** is in **Sunshine Canyon**.
Geo-Political Entity	GPE	countries, states	**Palo Alto** is raising the fees for parking.

Figure 8.5 A list of generic named entity types with the kinds of entities they refer to.

Citing high fuel prices, [ORG **United Airlines**] said [TIME **Friday**] it has increased fares by [MONEY **$6**] per round trip on flights to some cities also served by lower-cost carriers. [ORG **American Airlines**], a unit of [ORG **AMR Corp.**], immediately matched the move, spokesman [PER **Tim Wagner**] said. [ORG **United**], a unit of [ORG **UAL Corp.**], said the increase took effect [TIME **Thursday**] and applies to most routes where it competes against discount carriers, such as [LOC **Chicago**] to [LOC **Dallas**] and [LOC **Denver**] to [LOC **San Francisco**].

APPLICATIONS OF NER

Used as a step in many NLP applications, e.g.,

- In question-answering
- To determine consumer sentiment towards a particular entity
- To link to information in structured knowledge sources like Wikipedia
- To build semantic representations - extract events, and relationships between entities

NER CHALLENGES

- Entities can include multiple words - finding the boundary of an entity can be challenging (segmentation challenge, e.g., “river farming community”)
- Cross-type confusion:

[PER Washington] was born into slavery on the farm of James Burroughs.
[ORG Washington] went up 2 games to 1 in the four-game series.
Blair arrived in [LOC Washington] for what may well be his last state visit.
In June, [GPE Washington] passed a primary seatbelt law.

Figure 8.6 Examples of type ambiguities in the use of the name *Washington*.

SPAN TAGGING

- Standard approach: BIO tagging is used for span-recognition problems, and treats NER as a word-by-word sequence labeling task
  - Tokens that appear at the beginning of a span are labeled with B
  - Tokens that occur inside the span are labeled with I
  - Tokens outside of any span of interest are labeled with O
- BIOES tagging additionally has
  - A tag E for end of a span
  - A tag S for a span consisting of only 1 word
- IO variant doesn't have the B vs. I distinction and only uses I

Words	IO Label	BIO Label	BIOES Label
Jane	I-PER	B-PER	B-PER
Villanueva	I-PER	I-PER	E-PER
of	O	O	O
United	I-ORG	B-ORG	B-ORG
Airlines	I-ORG	I-ORG	I-ORG
Holding	I-ORG	I-ORG	E-ORG
discussed	O	O	O
the	O	O	O
Chicago	I-LOC	B-LOC	S-LOC
route	O	O	O
.	O	O	O

Figure 8.7 NER as a sequence model, showing IO, BIO, and BIOES taggings.

[PER **Jane Villanueva**] of [ORG **United**] , a unit of [ORG **United Airlines Holding**] , said the fare applies to the [LOC **Chicago**] route.

HMM POS

HIDDEN MARKOV MODEL (HMM) OVERVIEW

- HMM is a classic model that serves as the foundation for many AI approaches
- HMM is a probabilistic sequence model
  - Given a *sequence of items*, HMM computes a probability over the possible sequences of labels (and chooses the *best label sequence*)

MARKOV CHAINS

- A Markov chain describes the probabilities of a sequence of random variables (or states)
- Markov assumption: to predict the next step in a sequence, all you need to know is the current state
  - Markov Assumption: $P(q_i = a \,|\, q_1 \ldots q_{i-1}) = P(q_i = a \,|\, q_{i-1})$, where $q_1, q_2, \ldots, q_n$ are states and a is the probability of moving from state q_{i-1} to state q_i
  - The current state encapsulates all the information from the sequence up to this point (you don't need to inspect prior states)

Example

- Each arc specifies the transition probability of going from one state to another
- The sum of the transition probabilities out of a node is 1
- The example in b) in the figure is equivalent to a bigram language model - given a word, we have a list of words that can follow this word, along with their probabilities
- We can in fact use a Markov Chain of all vocabulary words and use it to compute the probability of any sequence

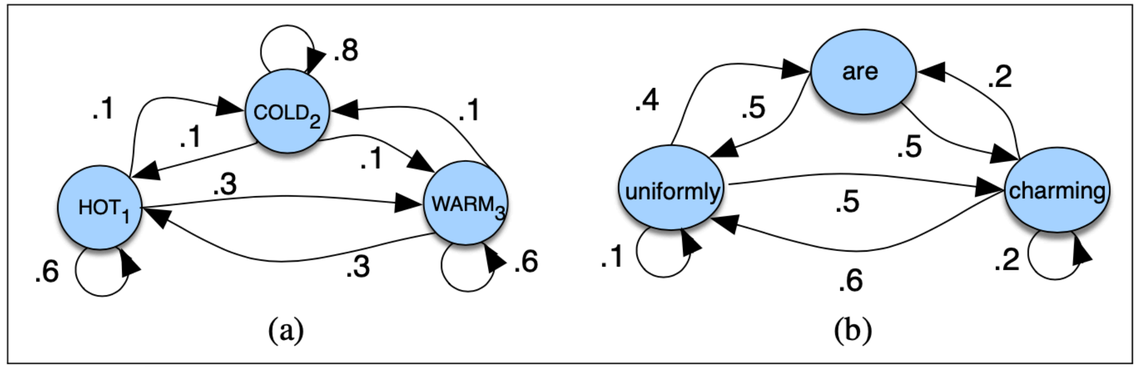

Figure 8.8 A Markov chain for weather (a) and one for words (b), showing states and transitions. A start distribution π is required; setting $\pi = [0.1, 0.7, 0.2]$ for (a) would mean a probability 0.7 of starting in state 2 (cold), probability 0.1 of starting in state 1 (hot), etc.

MARKOV CHAINS (CONT.)

- More formally, we have the following parameters that define a particular Markov chain:

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} a_{12} \ldots a_{N1} \ldots a_{NN}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
$\pi = \pi_1, \pi_2, \ldots, \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

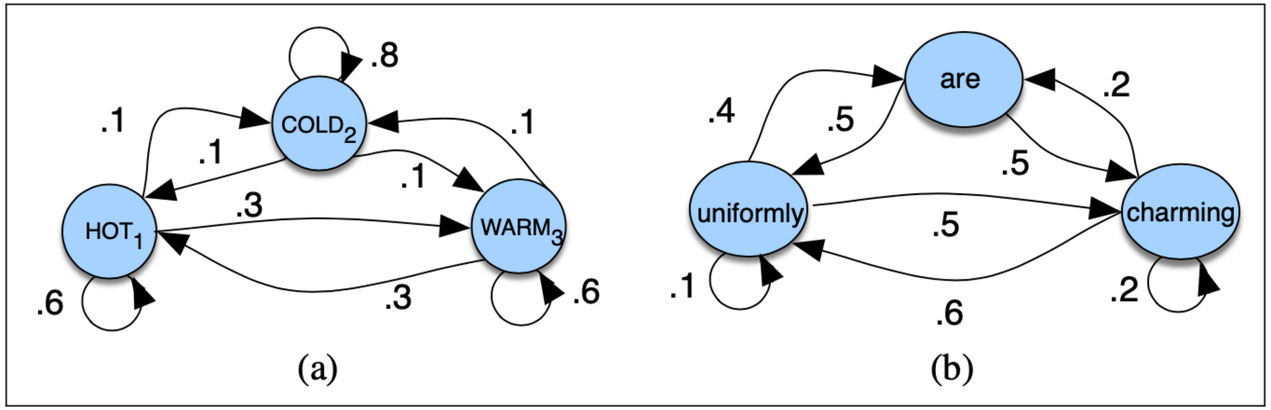

Figure 8.8 A Markov chain for weather (a) and one for words (b), showing states and transitions. A start distribution π is required; setting $\pi = [0.1, 0.7, 0.2]$ for (a) would mean a probability 0.7 of starting in state 2 (cold), probability 0.1 of starting in state 1 (hot), etc.

HIDDEN MARKOV MODEL

- A Markov Chain captures the probability for a sequence of observed events, e.g., words in the input
- PoS tags are not in the text itself but can be inferred from the text (they are hidden events)
- The hidden evens are considered "causal factors" in the HMM model

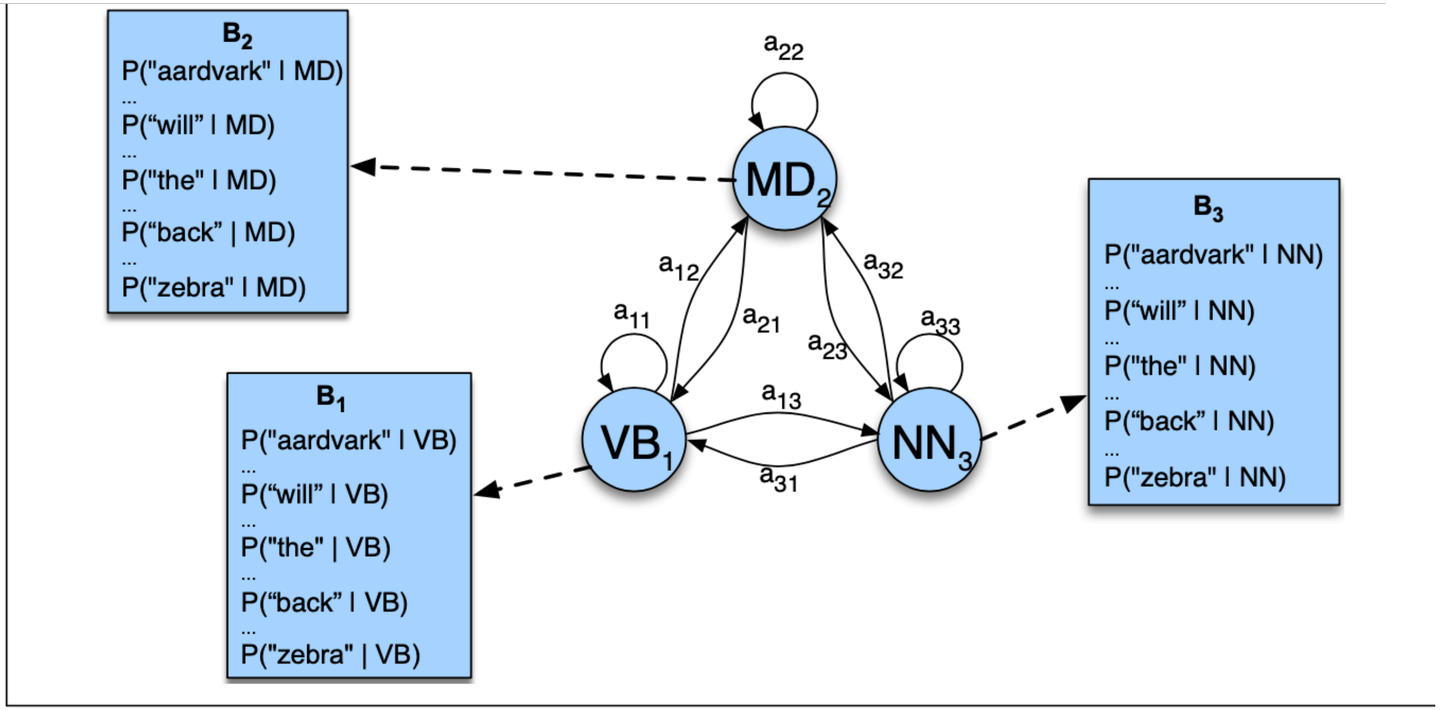

Figure 8.9 An illustration of the two parts of an HMM representation: the A transition probabilities used to compute the prior probability, and the B observation likelihoods that are associated with each state, one likelihood for each possible observation word.

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} \ldots a_{ij} \ldots a_{NN}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \ldots o_T$	a sequence of T **observations**, each one drawn from a vocabulary $V = v_1, v_2, \ldots, v_V$
$B = b_i(o_t)$	a sequence of **observation likelihoods**, also called **emission probabilities**, each expressing the probability of an observation o_t being generated from a state q_i
$\pi = \pi_1, \pi_2, \ldots, \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

FIRST-ORDER HIDDEN MARKOV MODEL

First-order HMM makes two assumptions:

- The probability of a particular state depends only on the previous state (Markov Assumption)

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

- The probability of an output observation, o_i, depends only on q_i, which is the state that produced the observation (does not depend on other observations or states) (Output Independence)

$$P(o_i \mid q_1, \ldots q_i, \ldots, q_T, o_1, \ldots, o_i, \ldots, o_T) = P(o_i \mid q_i)$$

HMM TAGGER COMPONENTS

- The A probability matrix stores the probabilities of a tag occurring given a previous tag:

$$P(t_i \mid t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

For example, in from the tagged WSJ corpus, we can compute the following:

$$P(VB \mid MD) = \frac{C(MD, VB)}{C(MD)} = \frac{10471}{13124} = 0.80$$

- The B probability matrix stores the probabilities that a tag would be associated with a given word:

$$P(w_i \mid t_i) = \frac{C(t_i, w_i)}{C(t_i)}$$

For example, in from the tagged WSJ corpus, we can compute the following:

$$P(will \mid MD) = \frac{C(MD, will)}{C(MD)} = \frac{4046}{13124} = 0.31$$

Interpretation: If we were to generate an MD, how likely is it that this modal would be the word “will”

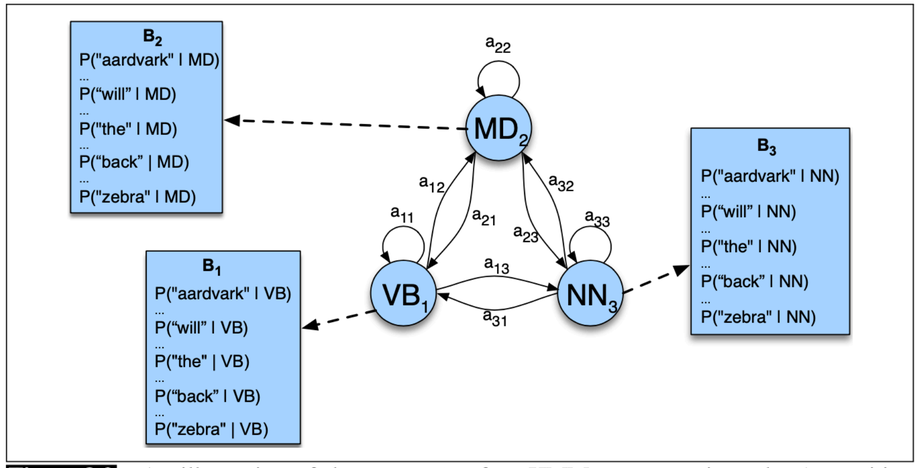

Figure 8.9 An illustration of the two parts of an HMM representation: the A transition probabilities used to compute the prior probability, and the B observation likelihoods that are associated with each state, one likelihood for each possible observation word.

HMM TAGGING AS A FORM OF DECODING

- Decoding

  The task for a model that contains hidden variables is typically to determine the sequence of hidden variables that correspond to the sequence of observations that have been made

- For HMM, the decoding task can be described as follows:

  Given an input HMM described by $\lambda = (A, B)$ and a sequence of observations $O = o_1, o_2, \ldots, o_T$, find the most probable sequence of states $Q = q_1 q_2 \ldots q_T$ (the states give us the tags)

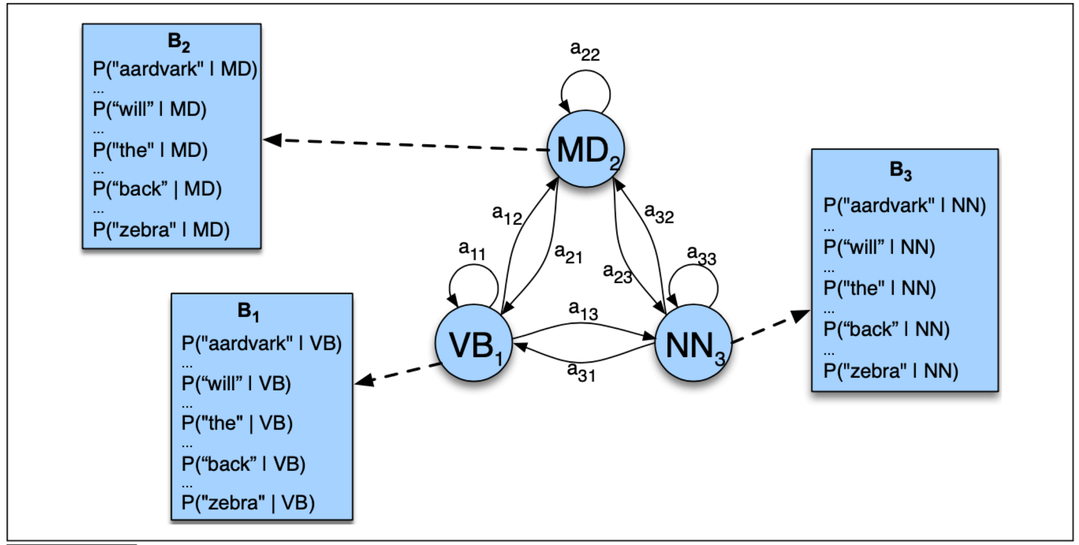

Figure 8.9 An illustration of the two parts of an HMM representation: the *A* transition probabilities used to compute the prior probability, and the *B* observation likelihoods that are associated with each state, one likelihood for each possible observation word.

HMM TAGGING IN NLP

- Choose the tag sequence $t_1 \dots t_n$ that is most probable given the observation sequence of words, $w_1 \dots w_n$:
  $\hat{t}_{1:n} = \arg\max_{t_1 \dots t_n} P(t_1 \dots t_n \mid w_1 \dots w_n)$
- Using the Bayes rule, this translates to: $\hat{t}_{1:n} = \arg\max_{t_1 \dots t_n} \frac{P(w_1 \dots w_n \mid t_1 \dots t_n) P(t_1 \dots t_n)}{P(w_1 \dots w_n)}$
- We apply two simplifying assumptions:
  1. The probability of a word appearing is independent of of neighboring words and tags (only depends on the word's own tag): $P(w_1 \dots w_n \mid t_1 \dots t_n) \approx \prod_{i=1}^{n} P(w_i \mid t_i)$
  2. The bigram assumption for tags – the probability of a tag is dependent only on the previous tag, rather than the whole tag sequence: $P(t_1 \dots t_n) \approx P(t_i \mid t_{i-1})$

This results in the following formulation: $\hat{t}_{1:n} = \arg\max_{t_1 \dots t_n} P(t_1 \dots t_n \mid w_1 \dots w_n) \approx \prod_{i=1}^{n} P(w_i \mid t_i) P(t_i \mid t_{i-1})$

Where $P(w_i \mid t_i)$ corresponds to the emission probability (matrix B), and $P(t_i \mid t_{i-1})$ corresponds to the transition probability (matrix A), where we can look up the various probabilities we need in order to compute the above product

EFFICIENT DECODING

- Uses the Viterbi algorithm
- Versions of Viterbi are also used for CRF

CONDITIONAL RANDOM FIELDS (CRFS)

SHORTCOMINGS OF HMM

- Adding features to help incorporate unknown words can be useful, e.g.
  - Capitalized words are likely to be proper nouns
  - Words ending with “-ed” are likely to be past tense of verbs
  - If the previous word is “the”, the current tag is unlikely to be a verb
- However, it is difficult to add arbitrary features to HMM
- We can however do this with a linear chain CRF (described next)

DIFFERENCE BETWEEN HMM AND CRF APPROACHES

- In HMM: if we have a sequence of input words $X = x_1 \dots x_n$ and we want to compute a sequence of output tags for them, $Y = y_1 \dots y_n$, we use Bayes and $P(X \mid Y)$, the likelihood of the words given the tags:

$$\begin{aligned}\hat{Y} &= \arg\max_{Y} p(Y \mid X) \\ &= \arg\max_{Y} p(X \mid Y) p(Y) \\ &= \arg\max_{Y} \prod_{i} p(x_i \mid y_) \prod_{i} p(y_i \mid y_{i-1})\end{aligned}$$

- In CRF, the posterior probability ($p(Y \mid X)$) is computed directly: $\hat{Y} = \arg\max_{Y \in \mathcal{Y}} P(Y \mid X)$

CRF APPROACH OVERVIEW

- CRF does not compute individual tag probabilities for each time step, but instead computes feature probabilities that get aggregated into a probability for the whole sequence
- CRF is a log-linear model that given an input sequence X, assigns a probability to a tag sequence Y, out of all possible sequences $\mathcal{Y}$
- CRF is like a multinomial logistic regression that works on sequences instead of individual data points

CRF APPROACH OVERVIEW (CONT.)

- Assuming we have K features, F_k, each with a weight w_k, CRF is expressed as follows

$$p(Y|X) = \frac{\exp\left(\sum_{k=1}^{K} w_k F_k(X,Y)\right)}{\sum_{Y' \in \mathcal{Y}} \exp\left(\sum_{k=1}^{K} w_k F_k(X,Y')\right)}$$

With the denominator often being represented as a function $Z(X)$:

$$Z(X) = \sum_{Y' \in \mathcal{Y}} \exp\left(\sum_{k=1}^{K} w_k F_k(X,Y')\right)$$

Which then leads to: $$p(Y|X) = \frac{1}{Z(X)} \exp\left(\sum_{k=1}^{K} w_k F_k(X,Y)\right)$$

- The K functions, $F_k(X, Y)$ are called global features as each is a property of the entire input sequence X and output sequence Y and are computed using local features f_k for each position i where each feature can use the previous and current output token, the whole input sequence and the current position i:

$$F_k(X, Y) = \sum_{i=1}^{n} f_k(y_{i-1}, y_i, X, i)$$

The restriction that only the current and previous output token can be used is what makes this a linear chain CRF which enables more efficient inference.

FEATURES IN A CRF TAGGER: KNOWN-WORD TEMPLATES

- In the previous slide, we stipulated that a local feature f_k at position i can use information from among y_{i-1}, y_i, X, i
- Assuming for simplicity that each CRF feature can take on the value of 0 or 1, consider the following example features:

$$\mathbb{1}\{x_i = \textit{the},\ y_i = \text{DET}\}$$
$$\mathbb{1}\{y_i = \text{PROPN}, x_{i+1} = \textit{Street}, y_{i-1} = \text{NUM}\}$$
$$\mathbb{1}\{y_i = \text{VERB}, y_{i-1} = \text{AUX}\}$$

y_{i-1} is the previous output token

y_i is the current output token

X is the whole input sequence

i the current position

- Features can be hand-crafted or populated through feature templates from the training set, e.g., we can count the number of instances from the training set that match the following template:

$$\langle y_i, x_i \rangle, \langle y_i, y_{i-1} \rangle, \langle y_i, x_{i-1}, x_{i+2} \rangle$$

For example, when considering the sentence Janet/NNP will/MD back/VB the/DT bill/NN, where x_i is the word *back*, the following features evaluating to 1 (i.e., True) will be generated (the feature subscripts are arbitrary in the example):

f_{3743}: y_i = VB and x_i = back — Janet/NNP will/MD back/VB the/DT bill/NN

f_{156}: y_i = VB and y_{i-1} = MD — Janet/NNP will/MD back/VB the/DT bill/NN

f_{99732}: y_i = VB and x_{i-1} = will and x_{i+2} = bill — Janet/NNP will/MD back/VB the/DT bill/NN

FEATURES IN A CRF TAGGER: WORD-SIGNATURE FEATURES

- "Word shape/signature" features can be used to capture unknown words
- They represent abstract letter patterns (akin to regular expressions), e.g., capture words that are 3-word abbreviations (eg., I.M.F.), or dates

x_i contains a particular prefix (perhaps from all prefixes of length ≤ 2)
x_i contains a particular suffix (perhaps from all suffixes of length ≤ 2)
x_i's word shape
x_i's short word shape

- For example, using the above feature templates on the word *well-dressed*, results in the following non-zero valued features:

prefix(x_i) = `w`
prefix(x_i) = `we`
suffix(x_i) = `ed`
suffix(x_i) = `d`
word-shape(x_i) = `xxxx-xxxxxxx`
short-word-shape(x_i) = `x-x`

FEATURES FOR CRF NAMED ENTITY RECOGNIZERS

- CRF features for NER are similar to those for PoS Tagging
- The following are used to lookup entities:
  - Gazetteer is a list of place names (e.g., www.geonames.org provides detailed geographical and political information)
  - Name-lists van used as entity dictionaries for names of corporations and products (e.g., from the US Census Bureau www.census.org)

identity of w_i, identity of neighboring words
embeddings for w_i, embeddings for neighboring words
part of speech of w_i, part of speech of neighboring words
presence of w_i in a **gazetteer**
w_i contains a particular prefix (from all prefixes of length ≤ 4)
w_i contains a particular suffix (from all suffixes of length ≤ 4)
word shape of w_i, word shape of neighboring words
short word shape of w_i, short word shape of neighboring words
gazetteer features

Figure 8.15 Typical features for a feature-based NER system.

- For example, assuming the named entity token "L'Occitane" does not appear on any lists, we would get the following non-zero valued feature values given our PoS template from earlier:

$\text{prefix}(x_i) = \texttt{L}$
$\text{prefix}(x_i) = \texttt{L'}$
$\text{prefix}(x_i) = \texttt{L'0}$
$\text{prefix}(x_i) = \texttt{L'0c}$
$\text{word-shape}(x_i) = \texttt{X'Xxxxxxxx}$
$\text{suffix}(x_i) = \texttt{tane}$
$\text{suffix}(x_i) = \texttt{ane}$
$\text{suffix}(x_i) = \texttt{ne}$
$\text{suffix}(x_i) = \texttt{e}$
$\text{short-word-shape}(x_i) = \texttt{X'Xx}$

NER EXAMPLE

- BIO labels can be used along with POS features in NER

[$_{PER}$ **Jane Villanueva**] of [$_{ORG}$ **United**] , a unit of [$_{ORG}$ **United Airlines Holding**] , said the fare applies to the [$_{LOC}$ **Chicago**] route.

Words	IO Label	BIO Label	BIOES Label
Jane	I-PER	B-PER	B-PER
Villanueva	I-PER	I-PER	E-PER
of	O	O	O
United	I-ORG	B-ORG	B-ORG
Airlines	I-ORG	I-ORG	I-ORG
Holding	I-ORG	I-ORG	E-ORG
discussed	O	O	O
the	O	O	O
Chicago	I-LOC	B-LOC	S-LOC
route	O	O	O
.	O	O	O

Figure 8.7 NER as a sequence model, showing IO, BIO, and BIOES taggings.

Words	POS	Short shape	Gazetteer	BIO Label
Jane	NNP	Xx	0	B-PER
Villanueva	NNP	Xx	1	I-PER
of	IN	x	0	O
United	NNP	Xx	0	B-ORG
Airlines	NNP	Xx	0	I-ORG
Holding	NNP	Xx	0	I-ORG
discussed	VBD	x	0	O
the	DT	x	0	O
Chicago	NNP	Xx	1	B-LOC
route	NN	x	0	O
.	.	.	0	O

Figure 8.16 Some NER features for a sample sentence, assuming that Chicago and Villanueva are listed as locations in a gazetteer. We assume features only take on the values 0 or 1, so the first POS feature, for example, would be represented as $\mathbb{1}\{\text{POS} = \text{NNP}\}$.

EVALUATION OF NAMED ENTITY RECOGNITION PERFORMANCE

- Typically we'd use Recall, Precision and F1
- Statistical significance testing can be used to evaluate if the differences in scores of two models are significant
- Each entity is considered as a unit (rather than the number of words), e.g., "United Airlines" being identified correctly counts as a single prediction
- Challenges:
  - A model that labeled Jane but not "Jane Villanueva" as a person will result in a false positive for O and a false negative for I-Per
  - We train on words but we test on multi-word entities, which is a mismatch between training and test conditions

Words	POS	Short shape	Gazetteer	BIO Label
Jane	NNP	Xx	0	B-PER
Villanueva	NNP	Xx	1	I-PER
of	IN	x	0	O
United	NNP	Xx	0	B-ORG
Airlines	NNP	Xx	0	I-ORG
Holding	NNP	Xx	0	I-ORG
discussed	VBD	x	0	O
the	DT	x	0	O
Chicago	NNP	Xx	1	B-LOC
route	NN	x	0	O
.	.	.	0	O

Figure 8.16 Some NER features for a sample sentence, assuming that Chicago and Villanueva are listed as locations in a gazetteer. We assume features only take on the values 0 or 1, so the first POS feature, for example, would be represented as $\mathbb{1}\{\text{POS} = \text{NNP}\}$.