Sequence Labeling

CS4742 Natural Language Processing
Lecture 08

Jiho Noh

Department of Computer Science
Kennesaw State University

CS4742 Summer 2025 [1]

KENNESAW STATE UNIVERSITY
COLLEGE OF COMPUTING AND SOFTWARE ENGINEERING

[1] This lecture is based on the slides from Dr. Hafiz Khan at KSU.

Sequence Labeling

- Sequence labeling assigns a label $y \in Y$ to each element in a given sequence x.
- If the input is a text sequence, elements are words or characters: $x = \{w_1, w_2, \ldots, w_n\}$, where n denotes the length of the text input.
- The objective is to find the best label sequence $y = \{y_1, y_2, \ldots, y_n\}$ for the input sequence x.
- **Algorithms**: Hidden Markov Model (HMM), Conditional Random Field (CRF), Recurrent Neural Network (RNN).

Sequence Labeling Tasks

- Y can be any set of labels. The most common tasks are:
  - **Parts of Speech (POS)** - noun, verb, pronoun, preposition, adverb, conjunction, participle, and article.
  - **Name Entity Recognition** - person, organization, location, date, time, money, etc.
- **Parts of Speech** and **Name entities** are useful clues for understanding sentence structure and meaning.
- Both are crucial for extracting meaningful information and improving the accuracy of NLP applications.

Part-of-Speech Tagging (POS Tagging)

POS tagging is the process of assigning a part of speech to each word in a sentence. The part of speech can be *noun*, *verb*, *adjective*, *adverb*, etc. A set of all POS tags is called the **tagset**, and various tagsets exist.

POS Tagging example:

(8.1) There/PRO/EX are/VERB/VBP 70/NUM/CD children/NOUN/NNS there/ADV/RB ./PUNC/.

(8.2) Preliminary/ADJ/JJ findings/NOUN/NNS were/AUX/VBD reported/VERB/VBN in/ADP/IN today/NOUN/NN 's/PART/POS New/PROPN/NNP England/PROPN/NNP Journal/PROPN/NNP of/ADP/IN Medicine/PROPN/NNP

Universal Dependencies (UD) Tag Set

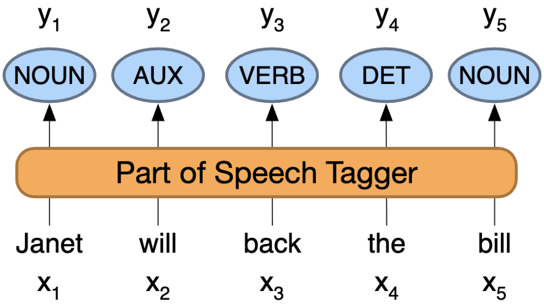
	Tag	Description	Example
Open Class	**ADJ**	Adjective: noun modifiers describing properties	*red*, *young*, *awesome*
	ADV	Adverb: verb modifiers of time, place, manner	*very*, *slowly*, *home*, *yesterday*
	NOUN	words for persons, places, things, etc.	*algorithm*, *cat*, *mango*, *beauty*
	VERB	words 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

Penn Treebank Tag Set

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*

POS Tagging

- **Tagging** is a *disambiguation* task; words are ambiguous.
- A word can have more than one *ambiguous* possible part-of-speech.
- The goal is to find the **correct** tag for the context.



Ambiguity Resolution

6 different parts of speech 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

Parts of Speech Tagging Performance

- The *accuracy* of part-of-speech tagging algorithms (the percentage of test set tags that match human gold labels) is extremely high.
- POS Tagging is known to be a **solved problem** in NLP.
- However, POS tagging can be found in various tasks:
  - Information retrieval, parsing, Text to Speech (TTS) applications, information extraction, linguistic research for corpora
  - POS can be used as an **intermediate step for higher level NLP tasks** – parsing, semantics analysis, translation etc.

Methods — First Take

Baseline: Maximum Likelihood

- Given an *ambiguous word*, choose the tag which is most frequent in the training corpus.
- $\arg\max_{\text{tag}} P(\text{tag}|\text{w}) = \arg\max_{\text{tag}} \frac{P(\text{tag,w})}{P(\text{w})} = \arg\max_{\text{tag}} P(\text{tag}, \text{w})$

Better Methods

Rule-based POS tagging:

- A set of handwritten rules and use *contextual information* to assign POS tags to words.
- **Example Rule:** if an ambiguous/unknown word ends with the suffix “ing” and is preceded by a word likely to be verb, tag it as a verb.
  - He is playing football.
  - *“Play”* – tag as **“Verb”**

Transformation based tagging:

- A pre-defined handcrafted rules combined with automatically rules.
- Automatic rules are generated from training data.

Even Better Methods

- **Deep learning models**: various deep models to tag *POS*
- **Probabilistic (stochastic) tagging**:
  - A stochastic approach computes frequency, probability from training set. During testing select highest probable tag.
  - Example algorithm: **Hidden Markov Model**

[2]

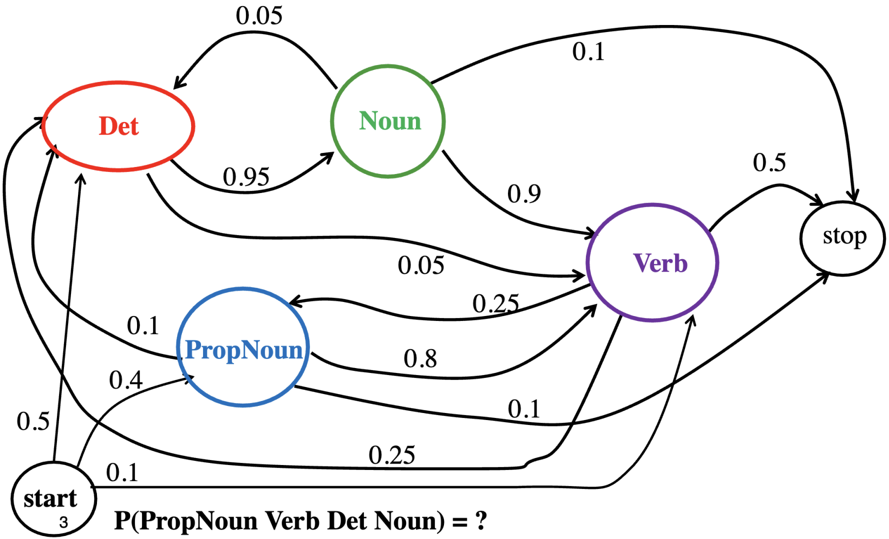
[2] Following slides are modified from Prof. Claire Cardie's slides and Prof. Raymond Mooney's slides. Some of the graphs are taken from the textbook.

POS Tagging using a Markov Model (Markov Chain)

- **Random variables** are used to represent the sequence of part-of-speech tags assigned to words in a sentence.
- With this model, each random variable corresponds to a word and takes on a value from the set of possible *POS tags*.
- A sequence of random variables $X_1, X_2, \ldots, X_n$ is said to be a **Markov chain** if the conditional probability of the next state depends only on the *current state* and not on the previous states (**Markov Assumption**).

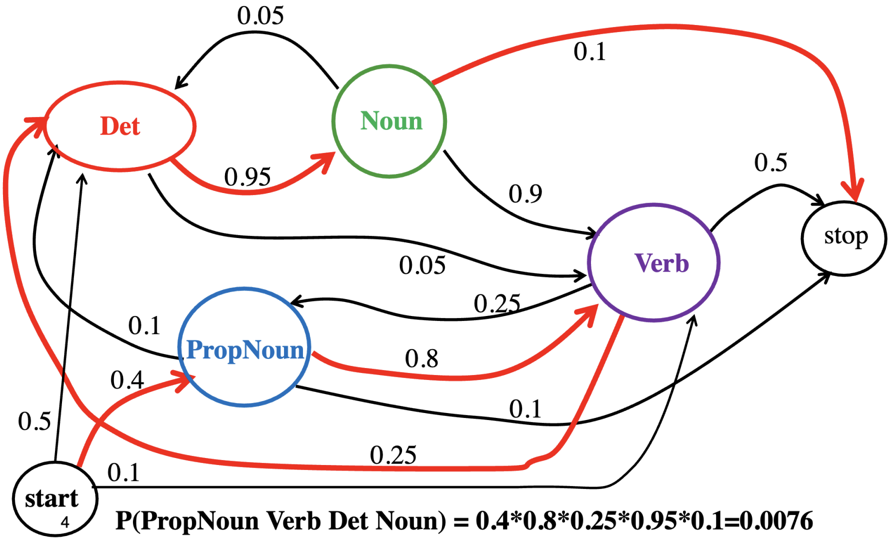
$$P(X_{n+1} = x | X_n = x_n, X_{n-1} = x_{n-1}, \ldots, X_1 = x_1) = P(X_{n+1} = x | X_n = x_n)$$

Sample Markov Model for POS



P(PropNoun Verb Det Noun) = ?

Sample Markov Model for POS



P(PropNoun Verb Det Noun) = 0.4*0.8*0.25*0.95*0.1=0.0076

Hidden Markov Model for POS Tagging

What do we know?

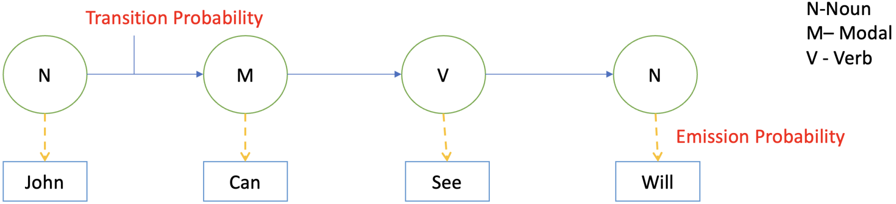
- **States** = POS tags
- **Transitions Probabilities** = probabilities of moving from one state to another

What do we not see in the previous figure?

- **Observation** = a sequence of words
- **Emission Probabilities** = probabilities of observing a word given a state (POS tag). Also called *observation likelihoods*.

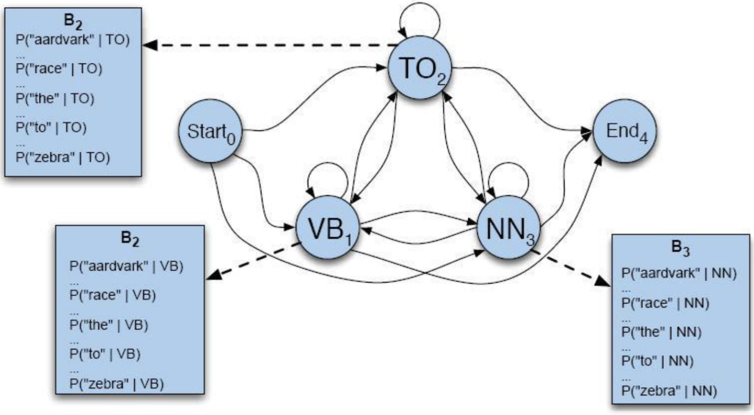
"**Hidden**" means the exact state sequence that generated the observations is not known.

HMM Probabilities



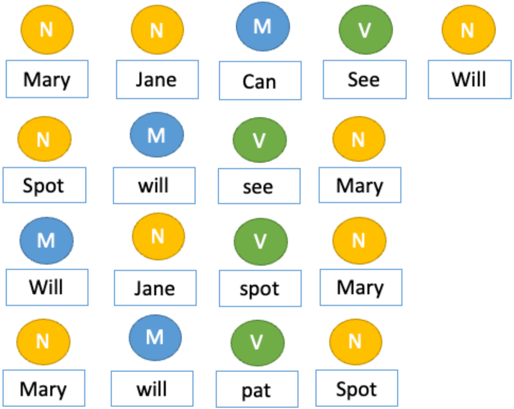
- HMM is a **generative model** that defines a joint probability distribution over the *observations* and *states*.
- Emmission probabilities (i.e., Observation likelihoods) generate the *observations* given the *states*.

Hidden Markov Model for POS Tagging



- **Hidden Markov Model (HMM)** represented as *finite state machine*.
- Note that in this representation, the number of nodes (states) = the size of the set of POS tags.

Estimation of Emission Probabilities



Example sentences:

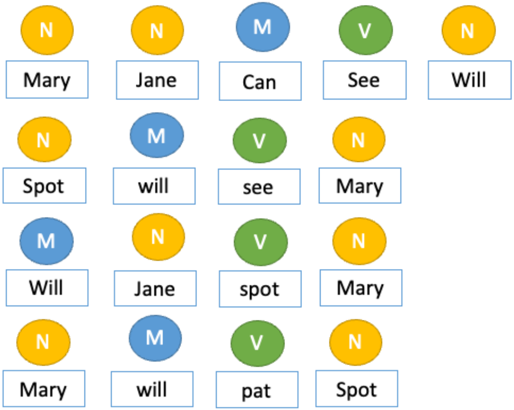
- *Mary Jane can see Will*
- *Spot will see Mary*
- *Will Jane spot Mary?*
- *Mary will pat Spot*

How to get Emission Probability?

Estimating of Emission Probabilities

Step 1: *Frequency count*

Word	N	M	V
Mary	4	0	0
Jane	2	0	0
Will	1	3	0
Spot	2	0	1
Can	0	1	0
See	0	0	2
pat	0	0	1
	9	4	4



Estimating of Emission Probabilities

Step 2: *Normalization (column wise)* $\rightarrow$ **emission probability**

Word	N	M	V
Mary	4	0	0
Jane	2	0	0
Will	1	3	0
Spot	2	0	1
Can	0	1	0
See	0	0	2
pat	0	0	1

Table: Raw Counts

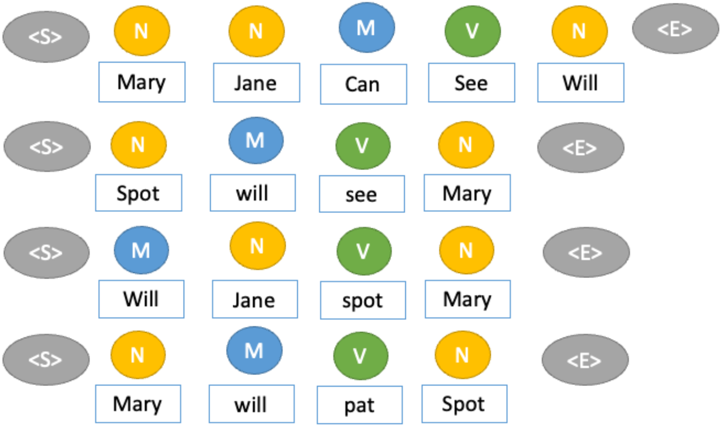
Word	N	M	V
Mary	4/9	0.00	0.00
Jane	2/9	0.00	0.00
Will	1/9	3/4	0.00
Spot	2/9	0.00	1/4
Can	0.00	1/4	0.00
See	0.00	0.00	2/4
pat	0.00	0.00	1/4

Table: Normalized

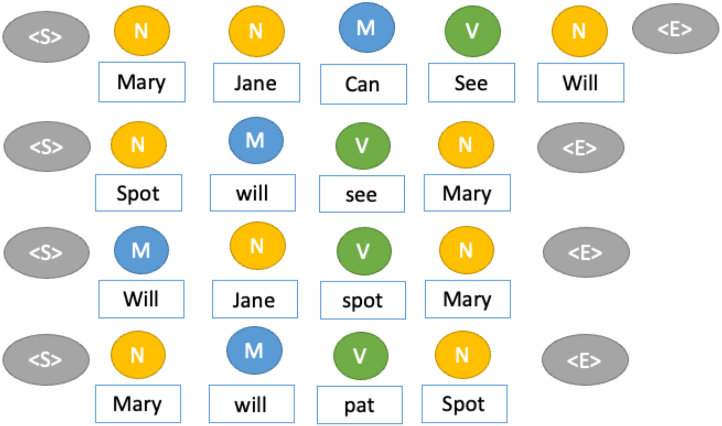
Estimating Transition Probabilities

Step 1: ***introduce beginning and end tags***

- Probability of next POS tag given previous POS tag is called **transition probability**.
- To indicate *start* and *end* of tag, we introduce two symbols <S>, <E> respectively.



Estimating Transition Probabilities



Step 2: ***Count co-occurrence of tags***

	N	M	V	<E>
<S>	3	1	0	0
N	1	3	1	4
M	1	0	3	0
V	4	0	0	0

Estimating Transition Probabilities

Step 3: ***Normalize (row wise)*** $\rightarrow$ **Transition Probability**

	N	M	V	<E>
<S>	3	1	0	0
N	1	3	1	4
M	1	0	3	0
V	4	0	0	0

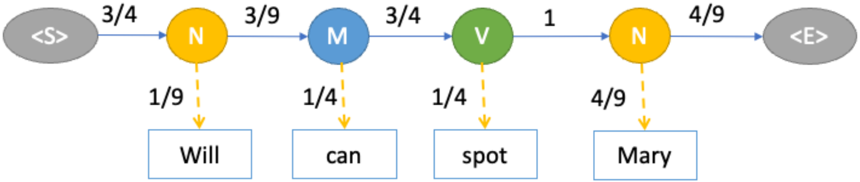
Table: Raw Counts

	N	M	V	<E>
<S>	3/4	1/4	0	0
N	1/9	3/9	1/9	4/9
M	1/4	0	3/4	0
V	4/4	0	0	0

Table: Normalized

Forward Algorithm

Determine the Likelihood of an Observation Sequence



Likelihood for this tag sequence:

$$3/4 \cdot 1/9 \cdot 3/9 \cdot 1/4 \cdot 3/4 \cdot 1/4 \cdot 1 \cdot 4/9 \cdot 4/9 = 0.0002572016$$

Viterbi Algorithm

HMM Decoding: Finds the *most likely sequence of states* (POS tags) that produced the observed sequence (words).

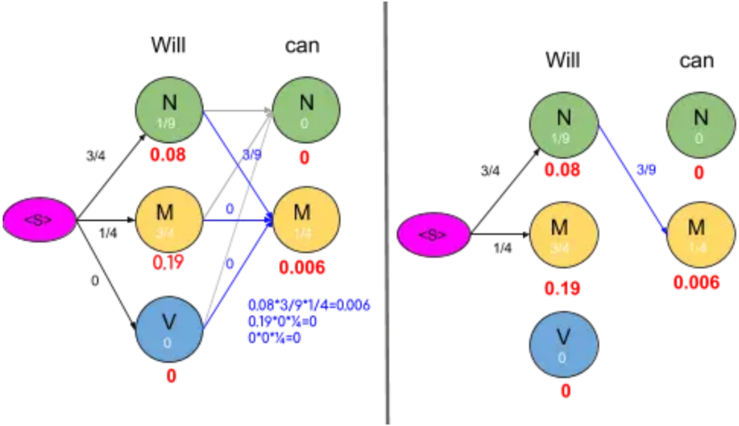
- Use transition and emission probability to predict next sequence.
- We don't want to search by enumerating all possible sequences of states. (Time complexity $O(N^T)$), where N is the number of states, T is the length of the sequence.
- **Dynamic Programming!**
- **Viterbi algorithm** has $O(N^2T)$ time complexity.

HMM Decoding: Viterbi Algorithm

Intuitions:

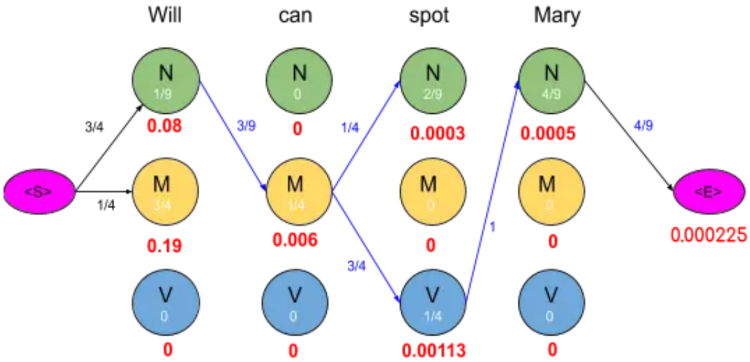
- The best path to a state at time t is the path that maximizes the probability to all the states at time $t-1$ times the transition probability to the state at time t.
- So, it is possible that the best path to a state at time $t-1$ is not the best path to the state at time t.
- We need a mechanism to find the best path to the final time step: backtracing.

HMM Tagging — Optimization



At each state, keep the **backpointer** to the previous state with the *highest probability*

HMM Tagging — Optimization



After the **forward computation**, a state will have only one incoming edge. Consequently, the backtracing will return the **best path of states** given the observation sequence.

function VITERBI(*observations* of len T, *state-graph* of len N) **returns** *best-path*

create a path probability matrix *viterbi[N+2,T]*

for each state s **from** 1 **to** N **do** ; initialization step

  $viterbi[s,1] \leftarrow a_{0,s} * b_s(o_1)$

  $backpointer[s,1] \leftarrow 0$

for each time step t **from** 2 **to** T **do** ; recursion step

 for each state s **from** 1 **to** N **do**

  $$viterbi[s,t] \leftarrow \max_{s'=1}^{N} viterbi[s',t-1] * a_{s',s} * b_s(o_t)$$

  $$backpointer[s,t] \leftarrow \operatorname*{argmax}_{s'=1}^{N} viterbi[s',t-1] * a_{s',s}$$

$$viterbi[q_F,T] \leftarrow \max_{s=1}^{N} viterbi[s,T] * a_{s,q_F}$$; termination step

$$backpointer[q_F,T] \leftarrow \operatorname*{argmax}_{s=1}^{N} viterbi[s,T] * a_{s,q_F}$$; termination step

return the backtrace path by following backpointers to states back in time from $backpointer[q_F,T]$

Viterbi Algorithm

3

[3] Slides adpated from CS447 UIUC https://courses.grainger.illinois.edu/cs447/fa2020/Slides/Lecture24.pdf

Name Entity Recognition (NER)

The task: find and classify names in text, for example:

*The European Commission [**ORG**] said on Thursday it disagreed with German [**MISC**] advice. Only France [**LOC**] and Britain [**LOC**] backed Fischler [**PER**] 's proposal.*

*"What we have to be extremely careful of is how other countries are going to take Germany's lead", Welsh National Farmers ' Union [**ORG**] (NFU [**ORG**]) chairman John Lloyd Jones [**PER**] said on BBC [**ORG**] radio.*

Named Entity Types

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	The Mt. Sanitas loop is in Sunshine Canyon.
Geo-Political Entity	GPE	countries, states, provinces	Palo Alto is raising the fees for parking.
Facility	FAC	bridges, buildings, airports	Consider the Golden Gate Bridge.
Vehicles	VEH	planes, trains, automobiles	It was a classic Ford Falcon.

These types were developed for the news domain as part of **NIST's Automatic Content Extraction (ACE) program**.
Other domains (e.g. *biomedical text*) require different types (proteins, genes, diseases, etc.)

NER on Word Sequences

NER on word sequences

We predict entities by *classifying words in context* and then extracting entities as **word sequences**.

Foreign	ORG	B-ORG
Ministry	ORG	I-ORG
spokesman	O	O
Shen	PER	B-PER
Guofang	PER	I-PER
told	O	O
Reuters	ORG	B-ORG
that	O	O
:	:	BIO encoding

Why NER is Hard?

"First National Bank Donates 2 Vans To Future School of Fort Smith"

- **Hard to work out boundaries of entity**
  - Is the first entity First National Bank or National Bank?
- **Hard to know if something is an entity**
  - Is there a school Future School or is it a future school?
- **Hard to know class of unknown/novel entity:**
  - To find out more about Zig Ziglar and read features by other Creators Syndicate writers
  - What class is Zig Ziglar? (A *Person*)
- **Entity class is ambiguous and depends on context**
  - Where Larry Ellison and Charles Schwab can live discreetly amongst wooded estates.
  - Charles Schwab is *PER* not *ORG* here!

Sequence Labeling Algorithms for NER

Statistical models:

- **Maximum Entropy Markov Models (MEMMs)**
- **Conditional Random Fields (CRFs)**

Neural models:

- *Recurrent networks* (or *transformers*) that predict a label at each time step, possibly with a **CRF output layer**.

Maximum Entropy Markov Models

MEMMs use a *logistic regression* (Maximum Entropy) classifier for each $P(t^{(i)} \mid w^{(i)}, t^{(i-1)})$.

$$P\left(t^{(i)} = t_k \mid t^{(i-1)}, w^{(i)}\right) = \frac{\exp\left(\sum_j \lambda_{jk} f_j\left(t^{(i-1)}, w^{(i)}\right)\right)}{\sum_l \exp\left(\sum_j \lambda_{jl} f_j\left(t^{(i-1)}, w^{(i)}\right)\right)}$$

Here, $t^{(i)}$: label of the *i-th* word vs. t_i: *i-th* label in the inventory.
This requires the definition of a **feature function** $f(t^{(i-1)}, w^{(i)})$ that returns an *n-dimensional feature vector* for predicting label $t^{(i)} = t_j$ given inputs $t^{(i-1)}$ and $w^{(i)}$.
Training returns weights λ_{jk} for each feature j used to predict label t_k.

Conditional Random Fields (CRFs)

Conditional Random Fields have the same mathematical definition as **MEMMs**, but:

- **CRFs** are trained *globally* to maximize the probability of the overall sequence,
- **MEMMs** are trained *locally* to maximize the probability of each individual label

This requires *dynamic programming*:

- **Training**: akin to the *Forward-Backward algorithm* used to train **HMMs** from unlabeled sequences
- **Decoding**: *Viterbi*

Feature-based NER (traditional approach)

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
base-phrase syntactic chunk label of w_i and 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)
w_i is all upper case
word shape of w_i, word shape of neighboring words
short word shape of w_i, short word shape of neighboring words
presence of hyphen

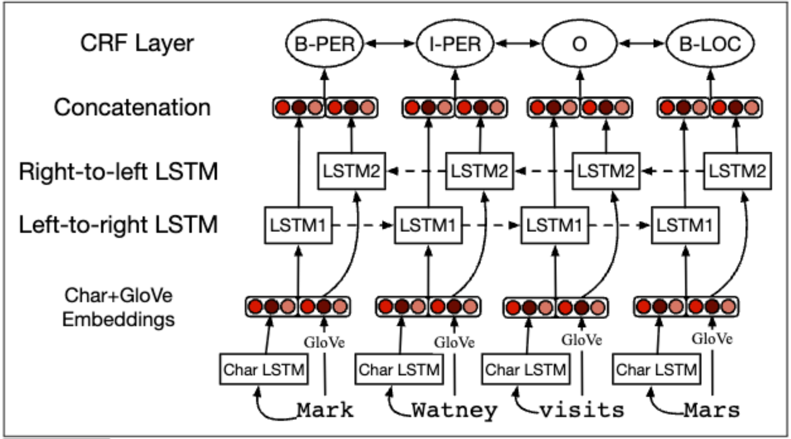
- **Train a sequence labeling model** (MEMM or CRF), using features such as the ones listed above for English.

Feature-based NER (Cont.)

- *Word Shape*: replace all upper-case letters with one symbol (e.g. "X"), all lower-case letters with another symbol ("x"), all digits with another symbol ("d"), and leave punctuation marks as is ("L'Occitane $\rightarrow$ "X'Xxxxxxxx").
- *Short Word Shape*: remove adjacent letters that are identical in word shape ("L'Occitane $\rightarrow$ "X'Xxxxxxxx" $\rightarrow$ "X'Xx").

Neural NER

Sequence RNN (e.g. *biLSTM* or *Transformer*) with a **CRF output layer**.
Input: word embeddings, possibly concatenated with character embeddings and other features, e.g.:



Rule-based NER

The textbook gives an example of an *iterative approach* that makes multiple passes over the text:

- **Pass 1**: Use **high-precision rules** to label (a small number of) unambiguous mentions
- **Pass 2**: Propagate the labels of the previously detected named entities to any mentions that are substrings (or acronyms?) of these entities
- **Pass 3**: Use *application-specific name lists* to identify further likely names (as *features*?)
- **Pass 4**: Now use a *sequence labeling approach* for NER, keeping the already labeled entities as **high-precision anchors**.

The basic ideas behind this approach (*label propagation*, using **high-precision items** as anchors) can be useful for other tasks as well.

Summary

Summary