LANGUAGE & AI: INFORMATION EXTRACTION

Chris Emmery
Department of Cognitive Science & AI
Tilburg University

@cmry • @_cmry • @cmry • cmry.github.io

♻️ RECAP PREVIOUS LECTURES

- We looked at how language might be **noisy** as input.
- We discussed several way to **represent** and **model** language, count or prediction-based.
- We looked at **classification** based on such representations, but still **global** representations.

Today we are finally considering ***sequentiality*** *in full.* 🥳

NLP FOR DATA SCIENCE

ℹ️ INFORMATION COMPONENTS

Meta-data-level:

- Historical context (time-bound).
- Social context (sender -> receiver).
- Normative context (upvotes).
- Geo-situated context (location).

Collection-level:

- Topics.
- Dialogues.
- Semantics.
- Paraphrases.

Document-level:

- Sentiment, emotions, sarcasm.
- Facts / deception.
- Category / topic.
- Niche: toxicity, medical, legal.

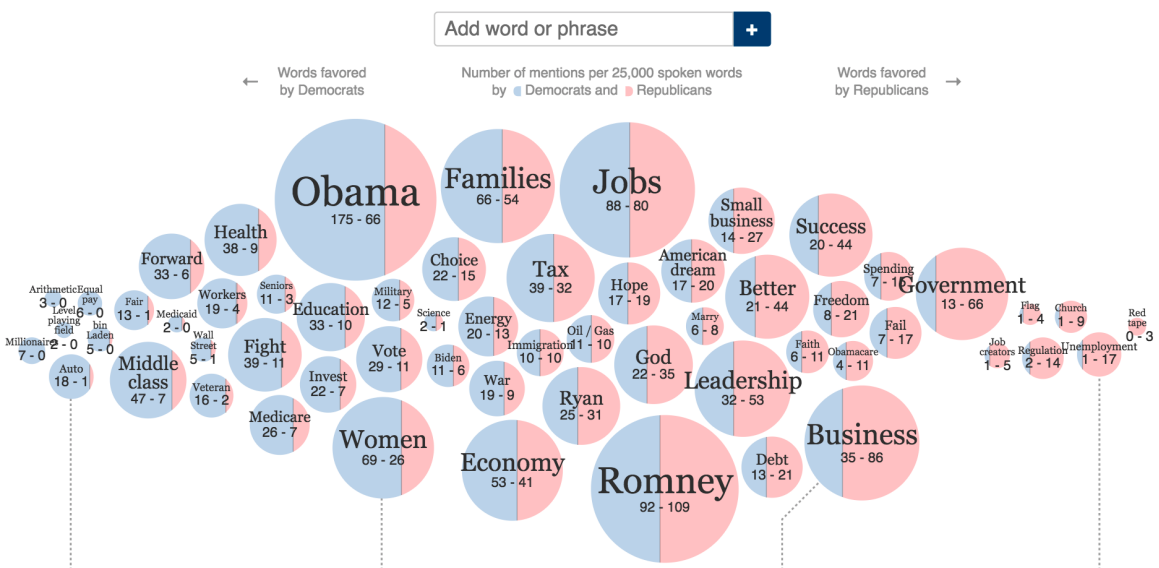
Text-level:

- Parsing.
- Disambiguation.
- Entities.
- Textual meta-data.

COUNTS

At the National Conventions, the Words They Used

A comparison of how often speakers at the two presidential nominating conventions used different words and phrases, based on an analysis of transcripts from the Federal News Service.



AUTO Democrats credited President Obama with the recovery of the auto industry after the 2009 bailout, while Republicans left the topic unmentioned.

WOMEN Democrats used the word much more frequently, primarily in reference to women's health and equal pay.

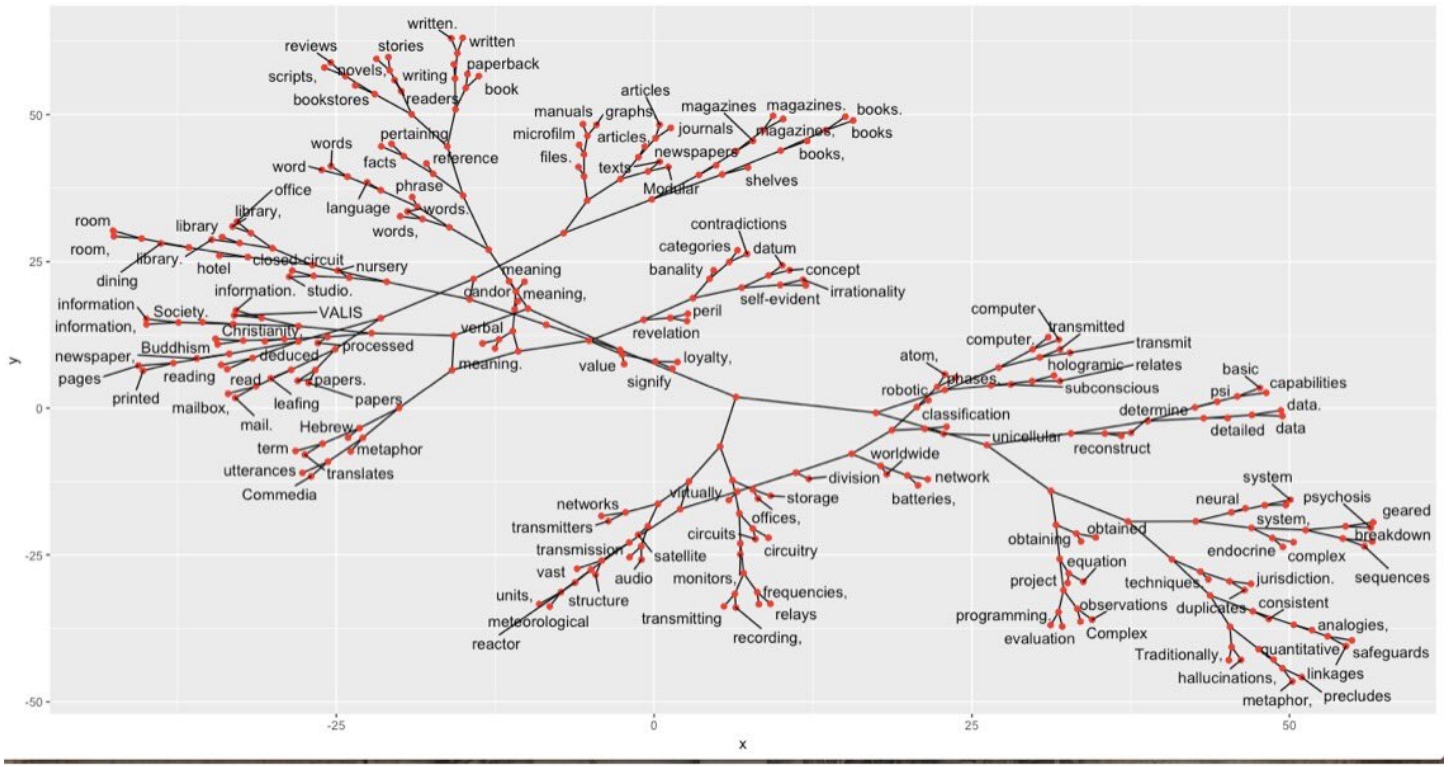
BUSINESS Republicans were more likely to talk about businesses, emphasizing Mr. Romney's private-sector experience and plans to improve the economy.

UNEMPLOYMENT Many Republican speakers brought up the still-high unemployment rate and the number of Americans who remain jobless, while Democrats largely avoided the topic.

TOPICS

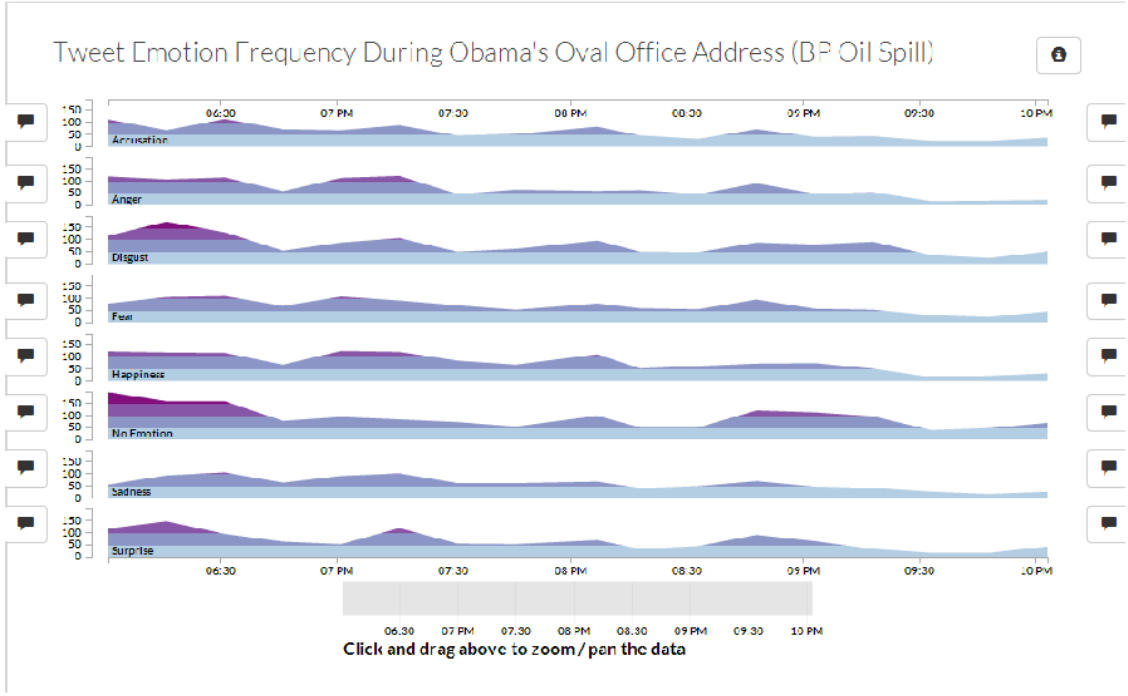
Termite: Visualization Techniques for Assessing Textual Topic Models (Chuang et al., 2012)

EMBEDDINGS


written.
written
reviews
stories
paperback
scripts,
novels,
writing
book
bookstores
readers
articles
manuals
graphs
magazines
magazines.
books.
microfilm
articles,
journals
magazines,
books
files.
newspapers
books,
texts
Modular
shelves
pertaining
words
reference
word
facts
phrase
office
language
words.
library,
words,
room
library
room,
library.
hotel
closed-circuit
nursery
dining
information.
studio.
information
Society.
VALIS
information,
Christianity
Buddhism
deduced
processed
newspaper
pages
reading
read
papers.
printed
mailbox,
leafing
papers
mail.
Hebrew
term
metaphor
utterances
translates
Commedia
contradictions
categories
datum
banality
concept
self-evident
irrationality
meaning
dandor
meaning,
peril
verbal
revelation
meaning.
value
loyalty,
signify
computer
computer,
transmitted
transmit
atom,
phases
holographic
relates
subconscious
robotic
basic
psi
capabilities
classification
determine
data.
unicellular
detailed
data
reconstruct
worldwide
division
network
batteries,
networks
virtually
storage
transmitters
offices,
circuits
transmission
satellite
circuitry
vast
audio
monitors,
units,
structure
frequencies,
meteorological
transmitting
relays
reactor
recording,
system
neural
psychosis
system,
geared
obtaining
obtained
breakdown
endocrine
complex
equation
project
techniques
jurisdiction.
sequences
observations
duplicates
consistent
programming,
Complex
analogies,
evaluation
Traditionally,
quantitative
safeguards
hallucinations,
linkages
metaphor,
precludes
50
25
0
-25
-50
y
-25
0
25
50
x

SENTIMENT

3 Visualization

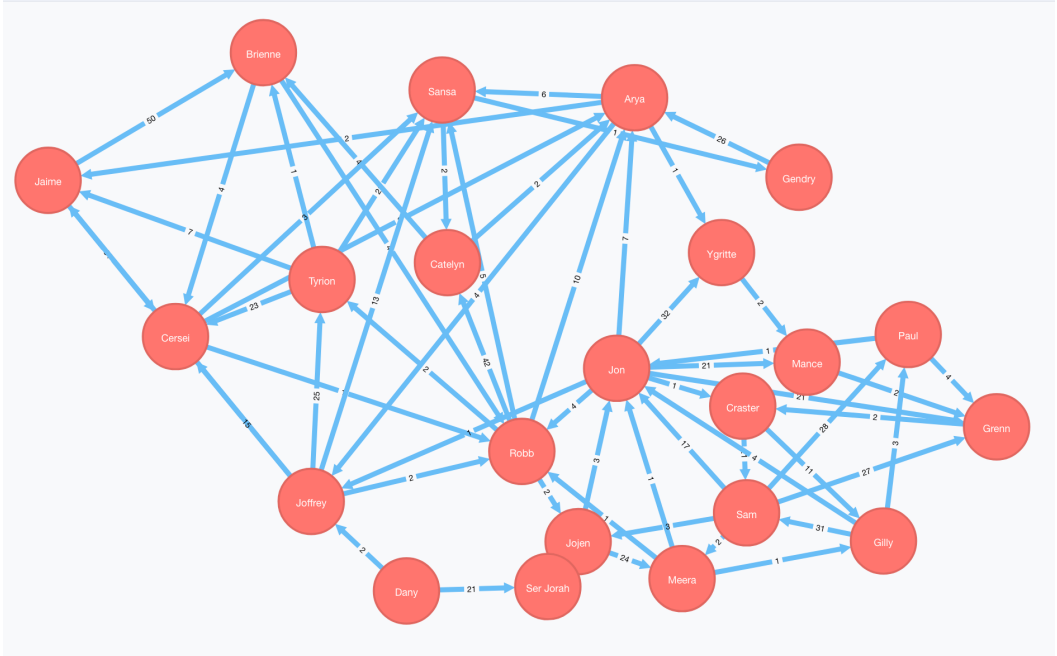


RELATIONS

```
$ MATCH p=()-[r:OCCURS_WITH]-() RETURN p ORDER BY r.freq DESC LIMIT 30
```

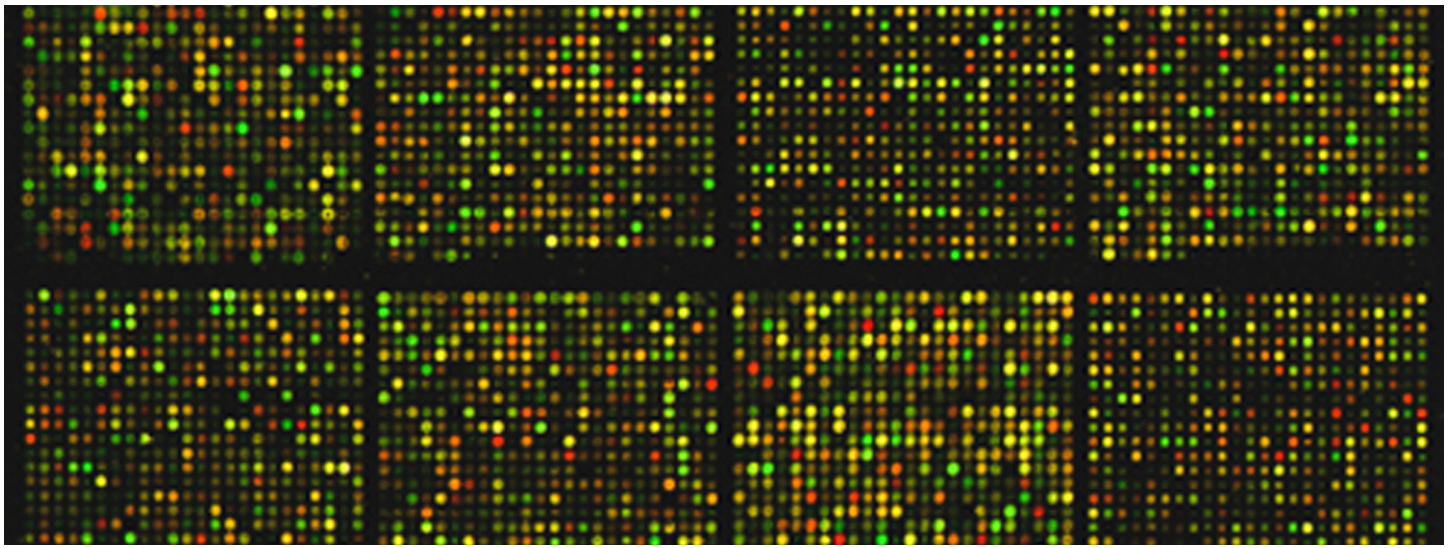
*(22) NER_Location(12) NER_Misc(4) NER_Organization(5) NER_Person(22) Tag(22)

*(59) OCCURS_WITH(59)



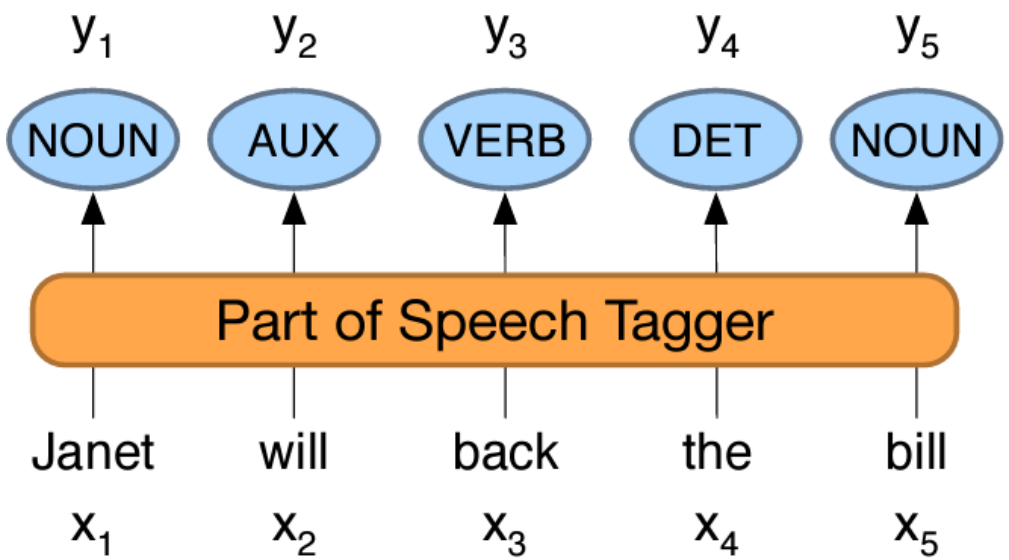
Displaying 22 nodes, 59 relationships (completed with 43 additional relationships).

SEQUENCE CLASSIFICATION



TASK

Given a sequence $S = x_1, x_2, \ldots, x_n$, predict a label **for each position**, i.e. $\hat{y} = y_1, y_2, \ldots, y_n$. Example, POS Tagging:



🚂 PART-OF-SPEECH (POS) TAGGING

- "Easy task": human 97%, various simple and complex algorithms perform the same.
- ~85% of word **types** are unambiguous, but 55 – 67% of **text** is ambiguous.
- Different tags for same word not equally likely.
- Majority baseline for common tree bank tasks: 92%.

🤷‍♀️ AMBIGUITY

- Adjective: earnings growth took a **back/JJ** seat.
- Noun: a small building in the **back/NN**.
- Verb (3d person singular): a clear majority of senators **back/VBP** the bill.
- Verb (base): Dave began to **back/VB** toward the door.
- Particle: enable the country to buy **back/RP** debt.
- Adverb: I was twenty-one **back/RB** then.

🏙 ADDITIONAL INFORMATION: ENTITIES

"[**ORG** United], a unit of [**ORG** UAL Corp.], said the increase of fares by [**MONEY** $6] 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]. Spokesman [**PER** Tim Wagner], cited high fuel prices in the [**GPE** United States] as a ..."

Useful for linking entities, information to entities (sentiment), extracting events, etc.

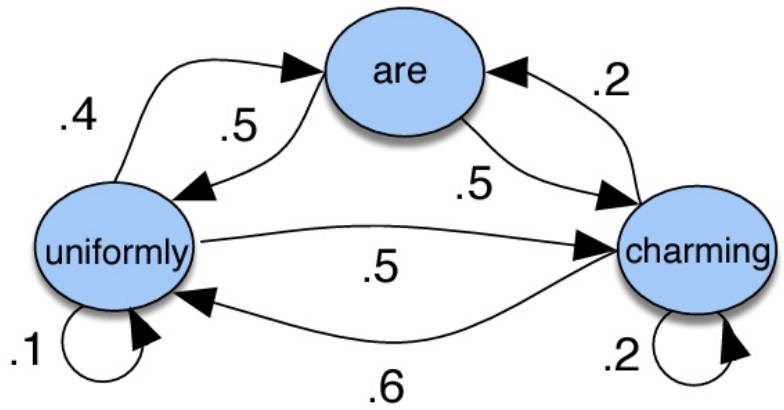
👀 NAMED ENTITY RECOGNITION (NER)

- Detect entity spans and assign labels.
- Most words are not named entities.
- Named entities can be ambiguous.
- Different **tagging** variants:

SEQUENCE TAGGERS

 MARKOV MODELS

- Only consider the **current** state to predict the **next**, not the **previous:** $P(q_i = a \mid q_1, \ldots, q_{i-1} = P(q_i = a \mid q_{i-1})$.
- Same as our bi-gram model! Edges are $P(w_i \mid w_j)$.
- Though, we have tags to predict per word, which are **hidden**.



HIDDEN MARKOV MODELS (HMMS)

We jointly model the *observed* and *hidden* events. We assume one causes the other. Formally:

$Q = q_1 q_2 \dots q_N$	a set of N **states**
$A = a_{11} \dots a_{ij} \dots a_{NN}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \dots o_T$	a sequence of T **observations**, each one drawn from a vocabulary $V = v_1, v_2, ..., v_V$
$B = b_i(o_t)$	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
$\pi = \pi_1, \pi_2, ..., \pi_N$	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, $\sum_{i=1}^{n} \pi_i = 1$

📊 HMMS AND MLE

Transition probabilities A:

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

Emission probabilities B:

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

"If we were going to generate a MD, how likely is it that this modal would be will?"

⛓ DECODING

Given as input an HMM $\lambda = (A, B)$ and a sequence of observations $O = o_1, o_2, \ldots, o_T$, find the most probable sequence of states $Q = q_1\ q_2\ q_3\ \ \ldots\ \ q_T$. This gives $\hat{t}$:

$$\hat{t}_{1:n} = \underset{t_1 \ldots t_n}{\operatorname{argmax}} P\left(t_1 \ldots t_n \mid w_1 \ldots w_n\right) \approx$$

$$\underset{t_1 \ldots t_n}{\operatorname{argmax}} \prod_{i=1}^{n} \overbrace{P\left(w_i \mid t_i\right)}^{\text{emission}} \overbrace{P\left(t_i \mid t_{i-1}\right)}^{\textit{transition}}$$

VITERBI

```
def viterbi(o, a, b, π):
    # getting T and N from actual input
    T, N = len(o), a.shape[0]
    viterbi = np.empty((N, T), 'd')
    backpointer = np.empty((N, T), 'B')

    # Vectorized init (no loop)
    viterbi[:, 0] = π * b[:, o[0]]
    backpointer[:, 0] = 0

    # Iterate throught the observations updating the tracking tables
    for t in range(1, T): # : = full s
        viterbi[:, t] = np.max(viterbi[:, t - 1] * a.T * b[np.newaxis, :, o[t]]
        backpointer[:, t] = np.argmax(viterbi[:, t - 1] * a.T, 1)

    bestpathpointer = np.empty(T, 'B')  # optimal model trajectory
    bestpathpointer[-1] = np.argmax(viterbi[:, T - 1])
    for t in reversed(range(1, T)):
        bestpathpointer[t - 1] = backpointer[t[i], i]

    return bestpathpointer  # unroll for bestpath
```

Based on Wikipedia pseudo-code implementation.

📝 EXAMPLE

	NNP	MD	VB	JJ	NN	RB	DT
<*s*>	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	0.0036
NN	0.0096	0.0176	0.0014	0.0086	0.1216	0.0177	0.0068
RB	0.0068	0.0102	0.1011	0.1012	0.0120	0.0728	0.0479
DT	0.1147	0.0021	0.0002	0.2157	0.4744	0.0102	0.0017

	Janet	will	back	the	bill
NNP	0.000032	0	0	0.000048	0
MD	0	0.308431	0	0	0
VB	0	0.000028	0.000672	0	0.000028
JJ	0	0	0.000340	0	0
NN	0	0.000200	0.000223	0	0.002337
RB	0	0	0.010446	0	0
DT	0	0	0	0.506099	0

!! SOME ISSUES

What to do with:

- Unknown words.
- Variety of proper names, new nouns and verbs.
- Can’t model those elegantly in a HMM.
- Log-linear models could but then they aren’t sequential.

CONDITIONAL RANDOM FIELDS

- Model $P(Y \mid X)$ directly (out of all possible tag sequences $\mathcal{Y}$), like LR but then massive.
- For K features, we use some function F_k to map it into a **global feature** vector to assign weight w_k.

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

- Global features $\rightarrow$ **local features**; use the previous output, the current one, the whole input, and the index:

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

LOCAL FEATURES

Typically constructed via templates:

$$\mathbb{1}\left(x_i = \text{the}, y_i = \text{DET}\right)$$

$$\mathbb{1}\left(y_i = \text{PROPN}, x_{i+1} = \text{Street}, y_{i-1} = \text{NUM}\right)$$

$$\mathbb{1}\left(y_i = \text{VERB}, y_{i-1} = \text{AUX}\right)$$

- Prefix / suffix.
- Word shapes.
- Gazetteer.
- Neighbor info.

INFERENCE FOR CRFS

Plug into Viterbi!

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i) \sum_{k=1}^{K} w_k f_k \left(y_{t-1}, y_t, X, t\right)$$

$$1 \leq j \leq N, 1 < t \leq T$$

Evaluate using standard metrics.

⛏ INFORMATION EXTRACTION

RELATION EXTRACTION

"[**ORG** United], a unit of [**ORG** UAL Corp.], said the increase of fares by [**MONEY** $6] 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]. Spokesman [**PER** Tim Wagner], cited high fuel prices in the [**GPE** United States] as a ..."

FINDING RDF TRIPLES

Structured information (e.g. Wikipedia) can be turned into (entity, relation, entity, or **Resource Description Framework** triples, e.g. (Evoluon, location, Eindhoven).

- Extract with *RegEx* (**Hearst patterns**): **LOC** [be]? (located|found|based|etc.) in **LOC** (expensive, in-domain).
- *Supervised* problem: relation Y/N filter, multi-class relation. Features: headwords, BoW, positional grams, NE types and scopes (labels expensive, brittle).
- **Distant Supervision or Bootstrap**: use known patterns to collect more web data. Confidence value to trade off hits (how many unique tules does it match) vs. finds (how many total tuples does it find).

⏳ (TEMPORAL) EVENT EXTRACTION

Absolute	Relative	Durations
April 24, 1916	yesterday	four hours
The summer of '77	next semester	three weeks
10:15 AM	two weeks from yesterday	six days
The 3rd quarter of 2006	last quarter	the last three quarters

Figure 17.11 Examples of absolute, relational and durational temporal expressions.

'United, a unit of UAL Corp., [**EVENT** said] [**EVENT** the increase] took effect Thursday and [**EVENT** applies] to most routes where it [**EVENT** competes] against discount carriers..."

Verbs, noun-phrases etc. Similar problems as NER. Extract temporal relations using supervised learning. Normalization rule-based. All difficult. 🙂