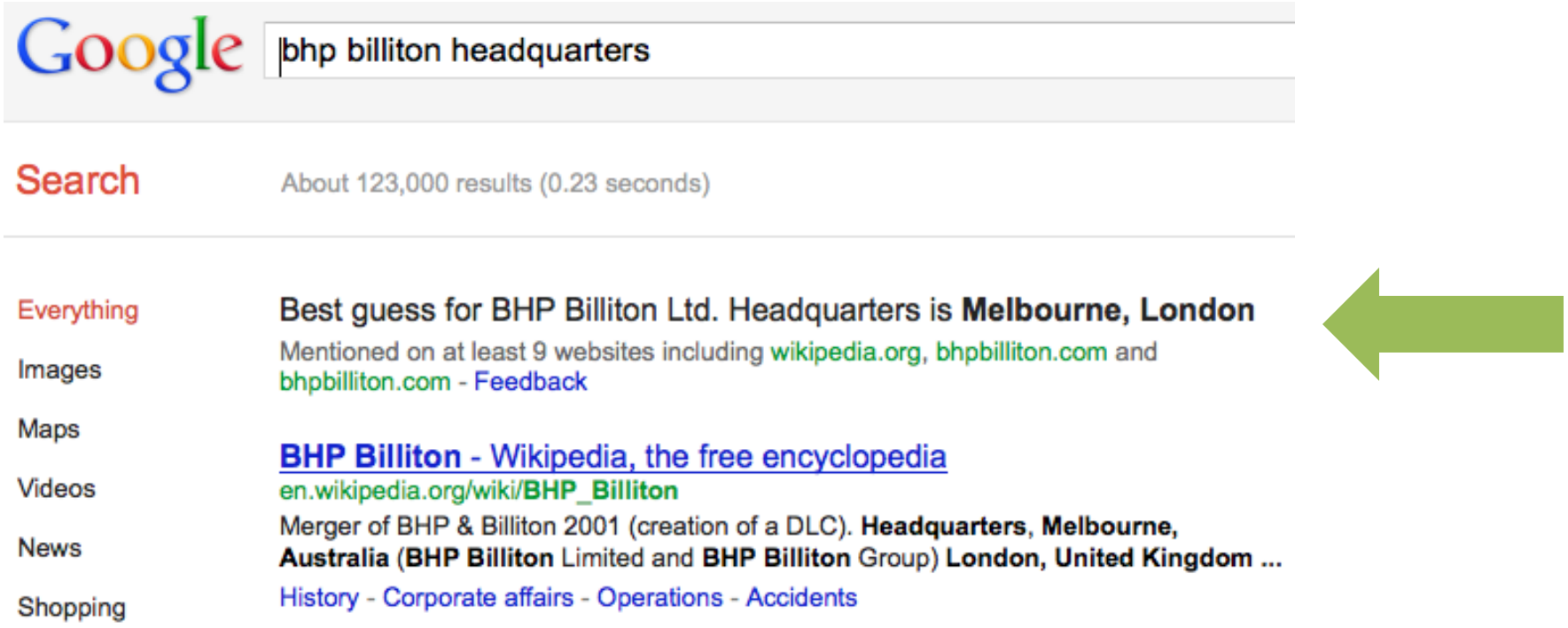
Sequence Labeling II
Named Entity Recognition with Max Entropy Markov Models

Mausam

(Slides based on Michael Collins, Heng Ji, Dan Jurafsky, Dan Klein, Chris Manning, Lev Ratinov, Luke Zettlemoyer)

Named Entity Recognition

Information Extraction



Named Entity Recognition (NER)

- A very important sub-task: find and classify names in text, for example:
  - The decision by the independent MP Andrew Wilkie to withdraw his support for the minority Labor government sounded dramatic but it should not further threaten its stability. When, after the 2010 election, Wilkie, Rob Oakeshott, Tony Windsor and the Greens agreed to support Labor, they gave just two guarantees: confidence and supply.

Named Entity Recognition (NER)

- A very important sub-task: find and classify names in text, for example:
  - The decision by the independent MP Andrew Wilkie to withdraw his support for the minority Labor government sounded dramatic but it should not further threaten its stability. When, after the 2010 election, Wilkie, Rob Oakeshott, Tony Windsor and the Greens agreed to support Labor, they gave just two guarantees: confidence and supply.

Named Entity Recognition (NER)

Person
Date
Location
Organi-
zation

- A very important sub-task: find and classify names in text, for example:
  - The decision by the independent MP Andrew Wilkie to withdraw his support for the minority Labor government sounded dramatic but it should not further threaten its stability. When, after the 2010 election, Wilkie, Rob Oakeshott, Tony Windsor and the Greens agreed to support Labor, they gave just two guarantees: confidence and supply.

Named Entity Recognition (NER)

- The uses:
  - Named entities can be indexed, linked off, etc.
  - Sentiment can be attributed to companies or products
  - A lot of IE relations are associations between named entities
  - For question answering, answers are often named entities.
- Concretely:
  - Many web pages tag various entities, with links to bio or topic pages, etc.
    - Reuters' OpenCalais, Evri, AlchemyAPI, Yahoo's Term Extraction, …
  - Apple/Google/Microsoft/… smart recognizers for document content

The Named Entity Recognition Task

Task: Predict entities in a text

Foreign	ORG
Ministry	ORG
spokesman	O
Shen	PER
Guofang	PER
told	O
Reuters	ORG
:	:

} Standard evaluation is per entity, *not* per token

Precision/Recall/F1 for IE/NER

- Recall and precision are straightforward for tasks like IR and text categorization, where there is only one grain size (documents)
- The measure behaves a bit funnily for IE/NER when there are *boundary errors* (which are *common*):
  - First Bank of Chicago announced earnings …
- This counts as both a fp and a fn
- Selecting *nothing* would have been better
- Some other metrics (e.g., MUC scorer) give partial credit (according to complex rules)

The ML Approach to NER

- Maximum Entropy Models (Borthwick, 1999; Chieu and Ng 2002; Florian et al., 2007)
- Decision Trees (Sekine et al., 1998)
- Class-based Language Model (Sun et al., 2002, Ratinov and Roth, 2009)
- Support Vector Machines (Takeuchi and Collier, 2002)
- Sequence Labeling Models
  - Hidden Markov Models (HMMs) (Bikel et al., 1997; Ji and Grishman, 2005)
  - Maximum Entropy Markov Models (MEMMs) (McCallum and Freitag, 2000)
  - Conditional Random Fields (CRFs) (McCallum and Li, 2003)

Sequence model approach to NER

Training

1. Collect a set of representative training documents
2. Label each token for its entity class or other (O)
3. Design feature extractors appropriate to the text and classes
4. Train a sequence classifier to predict the labels from the data

Testing

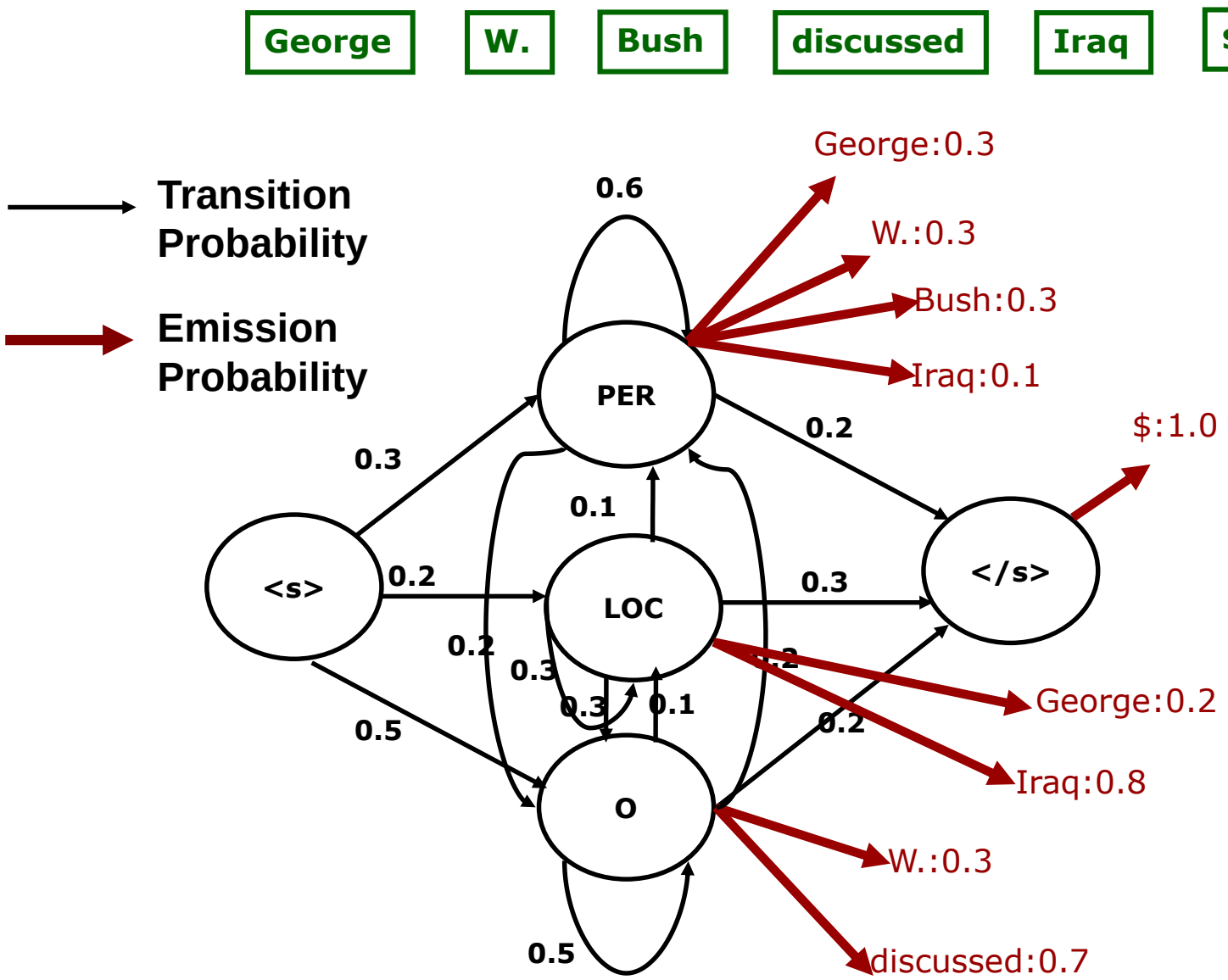
1. Receive a set of testing documents
2. Run sequence model inference to label each token
3. Appropriately output the recognized entities

Encoding classes for NER

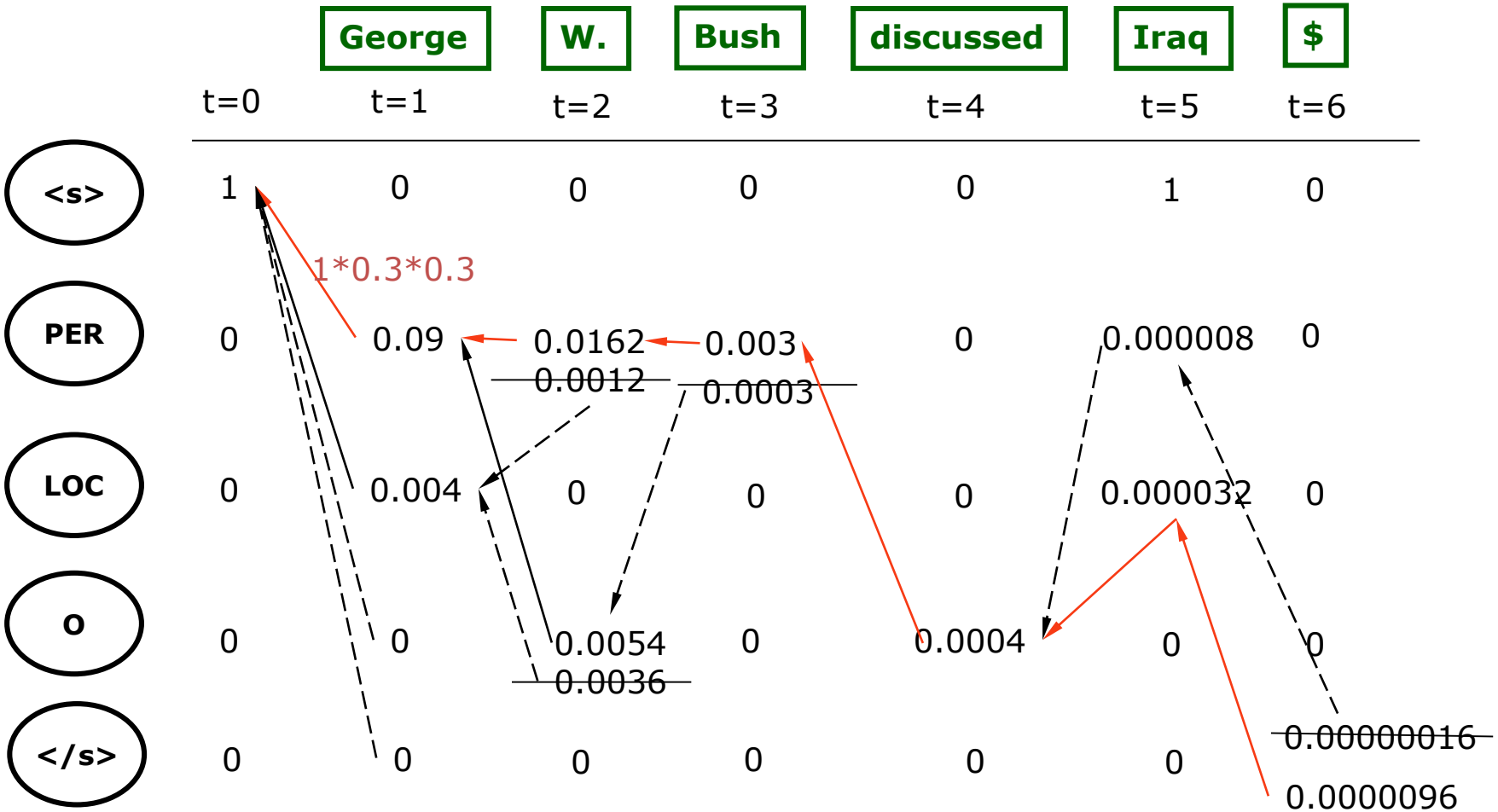
	IO encoding	IOB encoding
Fred	PER	B-PER
showed	O	O
Sue	PER	B-PER
Mengqiu	PER	B-PER
Huang	PER	I-PER
‘s	O	O
new	O	O
painting	O	O

Practically negligible differences in performance. IO much faster.

Markov Chain for a Simple Name Tagger



Viterbi Decoding of Name Tagger



Current = Previous * Transition * Emission

Limitations of HMMs

- Modeling more than necessary
  - joint probability distribution $p(y, x)$
- Assumes independent features
- Cannot represent overlapping features or long range dependences between observed elements
  - Need to enumerate all possible observation sequences
  - Very strict independence assumptions on the observations

- Toward discriminative/conditional models
  - Conditional probability P(label sequence y | observation sequence x) rather than joint probability P(y, x)
  - Allow arbitrary, non-independent features on the observation sequence X
  - The probability of a transition between labels may depend on past and future observations
  - Relax strong independence assumptions in generative models

Detour
Principle of Max Entropy

Principle of Maximum Entropy

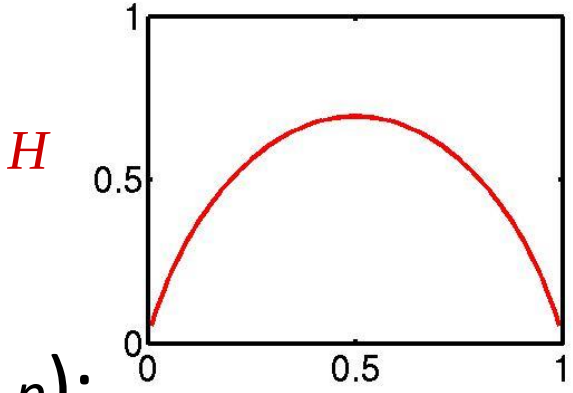
- Lots of distributions out there, most of them very spiked, specific, overfit.
- We want a distribution which is uniform except in specific ways we require.
- Uniformity means high entropy – we can search for distributions which have properties we desire, but also have high entropy.

Ignorance is preferable to error and he is less remote from the truth who believes nothing than he who believes what is wrong – Thomas Jefferson (1781)

(Maximum) Entropy

- Entropy: the uncertainty of a distribution.
- Quantifying uncertainty ("surprise"):
  - Event x
  - Probability p_x
  - "Surprise" $\log(1/p_x)$
- Entropy: expected surprise (over p):

$$\mathrm{H}(p) = E_p\left[\log_2 \frac{1}{p_x}\right] = -\sum_x p_x \log_2 p_x$$



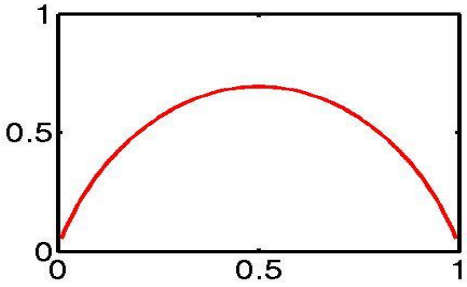
A coin-flip is most uncertain for a fair coin.

Maxent Examples I

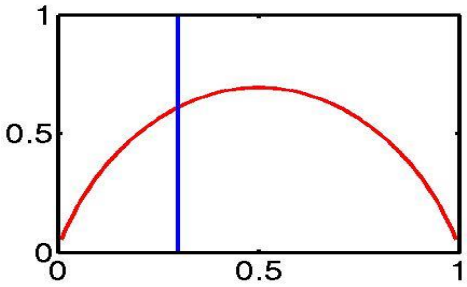
- What do we want from a distribution?
  - Minimize commitment = maximize entropy.
  - Resemble some reference distribution (data).
- Solution: maximize entropy H, subject to feature-based constraints:

$$E_p[f_i] = E_{\hat{p}}[f_i] \Leftrightarrow \sum_{x \in f_i} p_x = C_i$$

- Adding constraints (features):
  - Lowers maximum entropy
  - Raises maximum likelihood of data
  - Brings the distribution further from uniform
  - Brings the distribution closer to data

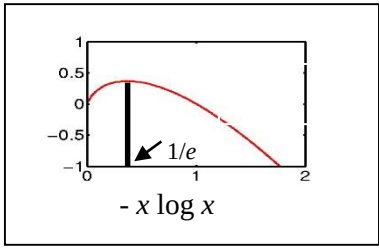


Unconstrained, max at 0.5

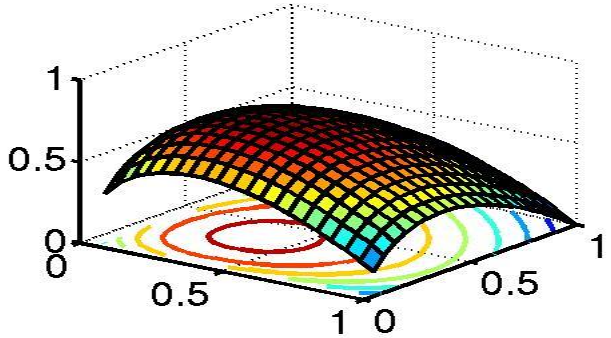


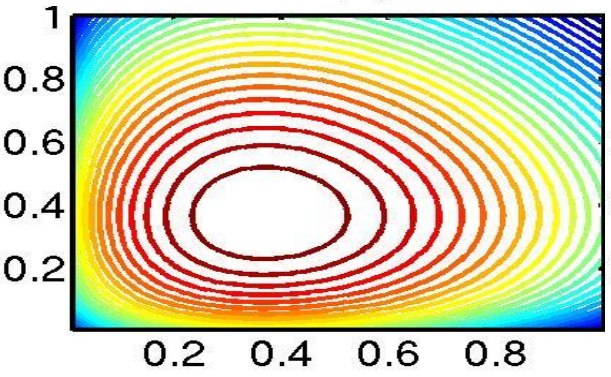
Constraint that $p_{\text{HEADS}} = 0.3$

Maxent Examples II

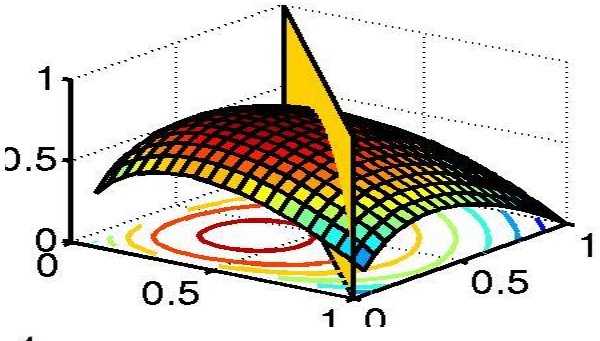


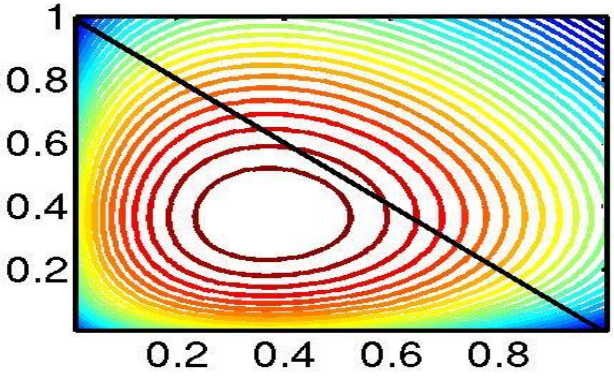
H(p_H p_T,)



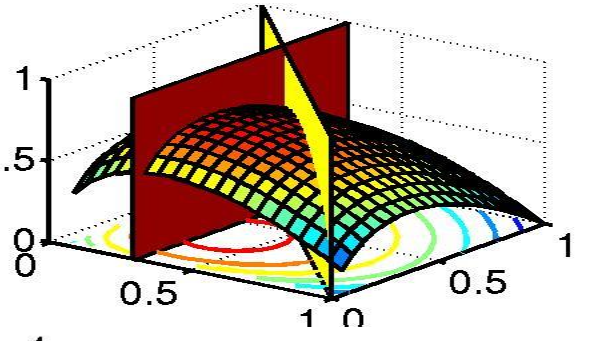


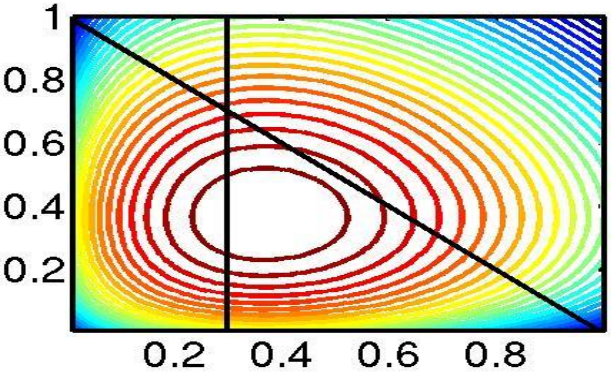
$p_H + p_T = 1$





$p_H = 0.3$





Maxent Examples III

- Let's say we have the following event space:

NN	NNS	NNP	NNPS	VBZ	VBD

- … and the following empirical data:

3	5	11	13	3	1

- Maximize H:

1/*e*	1/*e*	1/*e*	1/*e*	1/*e*	1/*e*

- … want probabilities: E[NN,NNS,NNP,NNPS,VBZ,VBD] = 1

1/6	1/6	1/6	1/6	1/6	1/6

Maxent Examples IV

- Too uniform!
- N* are more common than V*, so we add the feature f_N = {NN, NNS, NNP, NNPS}, with E[f_N] =32/36

NN	NNS	NNP	NNPS	VBZ	VBD
8/36	8/36	8/36	8/36	2/36	2/36

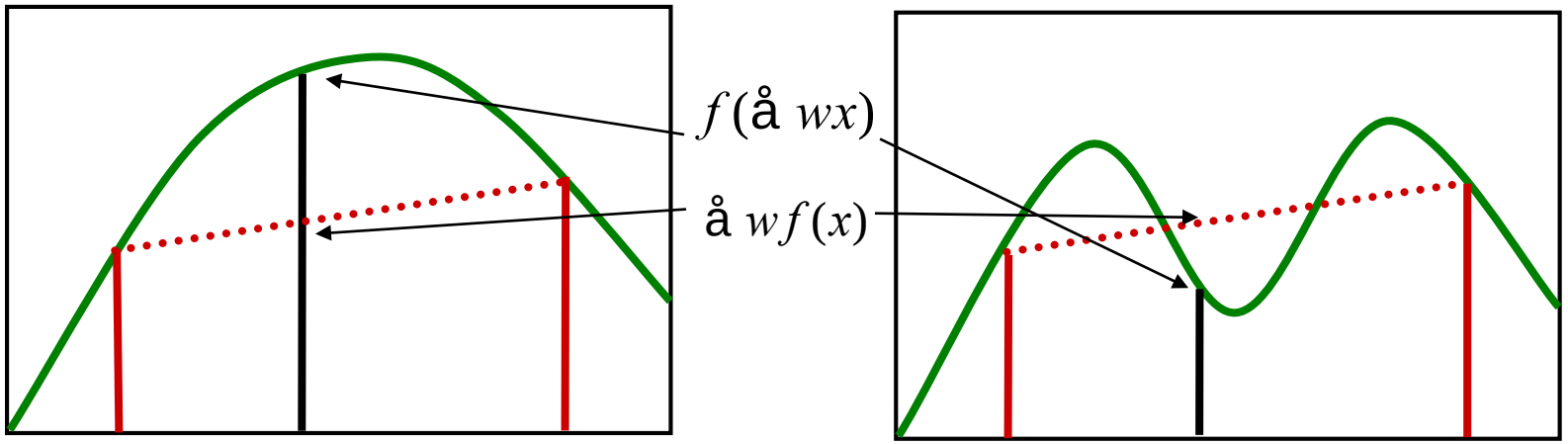
- … and proper nouns are more frequent than common nouns, so we add f_P = {NNP, NNPS}, with E[f_P] =24/36

4/36	4/36	12/36	12/36	2/36	2/36

- … we could keep refining the models, e.g., by adding a feature to distinguish singular vs. plural nouns, or verb types.

Convexity

$$f(\sum_i w_i x_i) \geq \sum_i w_i f(x_i) \quad \sum_i w_i = 1$$



Convex

Non-Convex

Convexity guarantees a single, global maximum because any higher points are greedily reachable.

Good news: Entropy is a convex function!

Theorem

- [Berger 96]

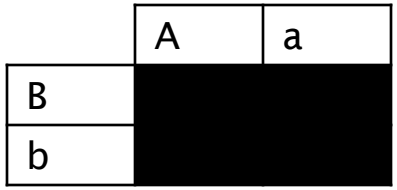
- Solution to max entropy optimization problem
- is equivalent to
- Probability distribution of multinomial logistic regression
  - whose weights maximize likelihood of data

Feature Overlap

- Maxent models handle overlapping features well.
- Unlike a NB model, there is no double counting!

Empirical

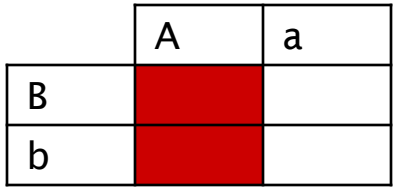
	A	a
B	2	1
b	2	1



All = 1

	A	a
B	1/4	1/4
b	1/4	1/4

	A	a
B		
b		



A = 2/3

	A	a
B	1/3	1/6
b	1/3	1/6

	A	a
B	λ_A	
b	λ_A	

A = 2/3

	A	a
B	1/3	1/6
b	1/3	1/6

	A	a
B	$\lambda'_A+\lambda''_A$	
b	$\lambda'_A+\lambda''_A$	

Example: Named Entity Feature Overlap

Grace is correlated with PERSON, but does not add much evidence on top of already knowing prefix features.

Local Context

	Prev	Cur	Next
State	Other	???	???
Word	at	Grace	Road
Tag	IN	NNP	NNP
Sig	x	Xx	Xx

Feature Weights

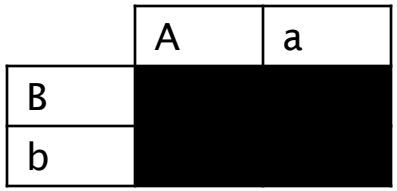
Feature Type	Feature	PERS	LOC
Previous word	*at*	-0.73	0.94
Current word	*Grace*	0.03	0.00
Beginning bigram	*<G*	0.45	-0.04
Current POS tag	NNP	0.47	0.45
Prev and cur tags	IN NNP	-0.10	0.14
Previous state	Other	-0.70	-0.92
Current signature	Xx	0.80	0.46
Prev state, cur sig	O-Xx	0.68	0.37
Prev-cur-next sig	x-Xx-Xx	-0.69	0.37
P. state - p-cur sig	O-x-Xx	-0.20	0.82
…			
Total:		**-0.58**	**2.68**

Feature Interaction

- Maxent models handle overlapping features well, but do not automatically model feature interactions.

Empirical

	A	a
B	1	1
b	1	0



All = 1

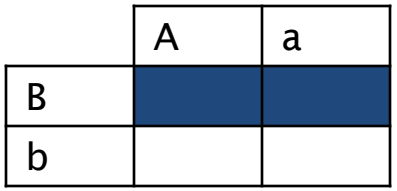
	A	a
B	1/4	1/4
b	1/4	1/4

	A	a
B	0	0
b	0	0

A = 2/3

	A	a
B	1/3	1/6
b	1/3	1/6

	A	a
B	λ_A	
b	λ_A	



B = 2/3

	A	a
B	4/9	2/9
b	2/9	1/9

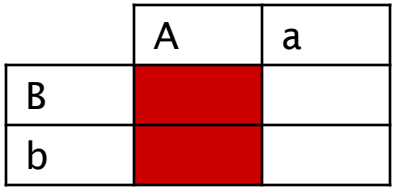
	A	a
B	$\lambda_A+\lambda_B$	λ_B
b	λ_A	

Feature Interaction

- If you want interaction terms, you have to add them:

Empirical

	A	a
B	1	1
b	1	0



	A	a
B		
b		

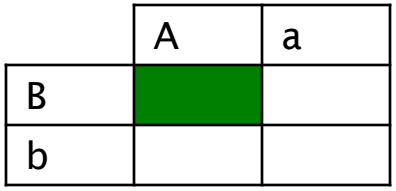
A = 2/3

	A	a
B	1/3	1/6
b	1/3	1/6

	A	a
B		
b		

B = 2/3

	A	a
B	4/9	2/9
b	2/9	1/9



	A	a
B		
b		

AB = 1/3

	A	a
B	1/3	1/3
b	1/3	0

- A disjunctive feature would also have done it (alone):

	A	a
B		
b		

	A	a
B	1/3	1/3
b	1/3	0

Feature Interaction

- For loglinear/logistic regression models in statistics, it is standard to do a greedy stepwise search over the space of all possible interaction terms.
- This combinatorial space is exponential in size, but that's okay as most statistics models only have 4–8 features.

- In NLP, our models commonly use hundreds of thousands of features, so that's not okay.
- Commonly, interaction terms are added by hand based on linguistic intuitions.

Example: NER Interaction

Previous-state and current-signature have interactions, e.g. P=PERS-C=Xx indicates C=PERS much more strongly than C=Xx and P=PERS independently.

This feature type allows the model to capture this interaction.

Local Context

	Prev	Cur	Next
State	Other	???	???
Word	at	Grace	Road
Tag	IN	NNP	NNP
Sig	x	Xx	Xx

Feature Weights

Feature Type	Feature	PERS	LOC
Previous word	*at*	-0.73	0.94
Current word	*Grace*	0.03	0.00
Beginning bigram	*<G*	0.45	-0.04
Current POS tag	NNP	0.47	0.45
Prev and cur tags	IN NNP	-0.10	0.14
Previous state	Other	-0.70	-0.92
Current signature	Xx	0.80	0.46
Prev state, cur sig	O-Xx	0.68	0.37
Prev-cur-next sig	x-Xx-Xx	-0.69	0.37
P. state - p-cur sig	O-x-Xx	-0.20	0.82
…			
Total:		**-0.58**	**2.68**

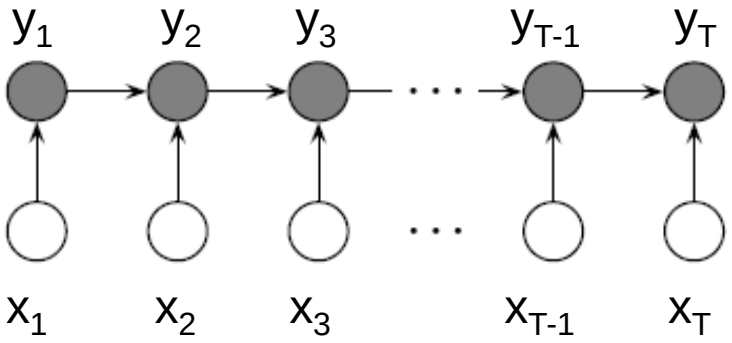
Why ME?

- Advantages
  - Combine multiple knowledge sources
    - Local
      - Word prefix, suffix, capitalization (POS - *(Ratnaparkhi, 1996)*)
      - Word POS, POS class, suffix (WSD - *(Chao & Dyer, 2002)*)
      - Token prefix, suffix, capitalization, abbreviation (Sentence Boundary - *(Reynar & Ratnaparkhi, 1997)*)
    - Global
      - N-grams *(Rosenfeld, 1997)*
      - Word window
      - Document title *(Pakhomov, 2002)*
      - Structurally related words *(Chao & Dyer, 2002)*
      - Sentence length, conventional lexicon *(Och & Ney, 2002)*
  - Combine *dependent* knowledge sources

Maximum Entropy *Markov* Models

Maximum Entropy Markov Models (MEMMs)

- A conditional model that representing the probability of reaching a state given an observation and the previous state
- Consider observation sequences to be events to be conditioned upon.



- Have all the advantages of Conditional Models
- No longer assume that features are independent

Bigram MEMMs

$$P(\vec{y} \mid \vec{x}) = \prod_{t=1}^{T+1} p(y_t \mid y_{[0..t-1]}, x_{[0..T]})$$

$$= \prod_{t=1}^{T+1} p(y_t \mid y_{t-1}, x_{[0..T]})$$

- y_0=<s>, y_{T+1}=</s>
- Each tag only depends on previous tag

Trigram MEMMs

$$P(\vec{y} \mid \vec{x}) = \prod_{t=1}^{T+1} p(y_t \mid y_{[0..t-1]}, x_{[0..T]})$$

$$= \prod_{t=1}^{T+1} p(y_t \mid y_{t-1}, x_{[0..T]})$$

$$= \prod_{t=1}^{T+1} p(y_t \mid y_{t-1}, y_{t-2}, x_{[0..T]})$$

- y_0=<s>, y_{-1}=<s>, y_{T+1}=</s>
- Each tag depends on previous two tags

Features for Sequence Labeling

Representation: Histories

- US/L president/O Obama/P visited/O Delhi/L to/O meet/O with/O Narendra/P Modi/P.

- Define: History -- a 4 tuple < y_{-2}, y_{-1}, $x_{[1..T]}$, i>
  - y_{-2}, y_{-1}: two previous tags
  - $x_{[1..T]}$: all words in the sentence
  - i: index of the word being tagged

- Example: History(Obama)
  - <L,O,US…Modi,3>

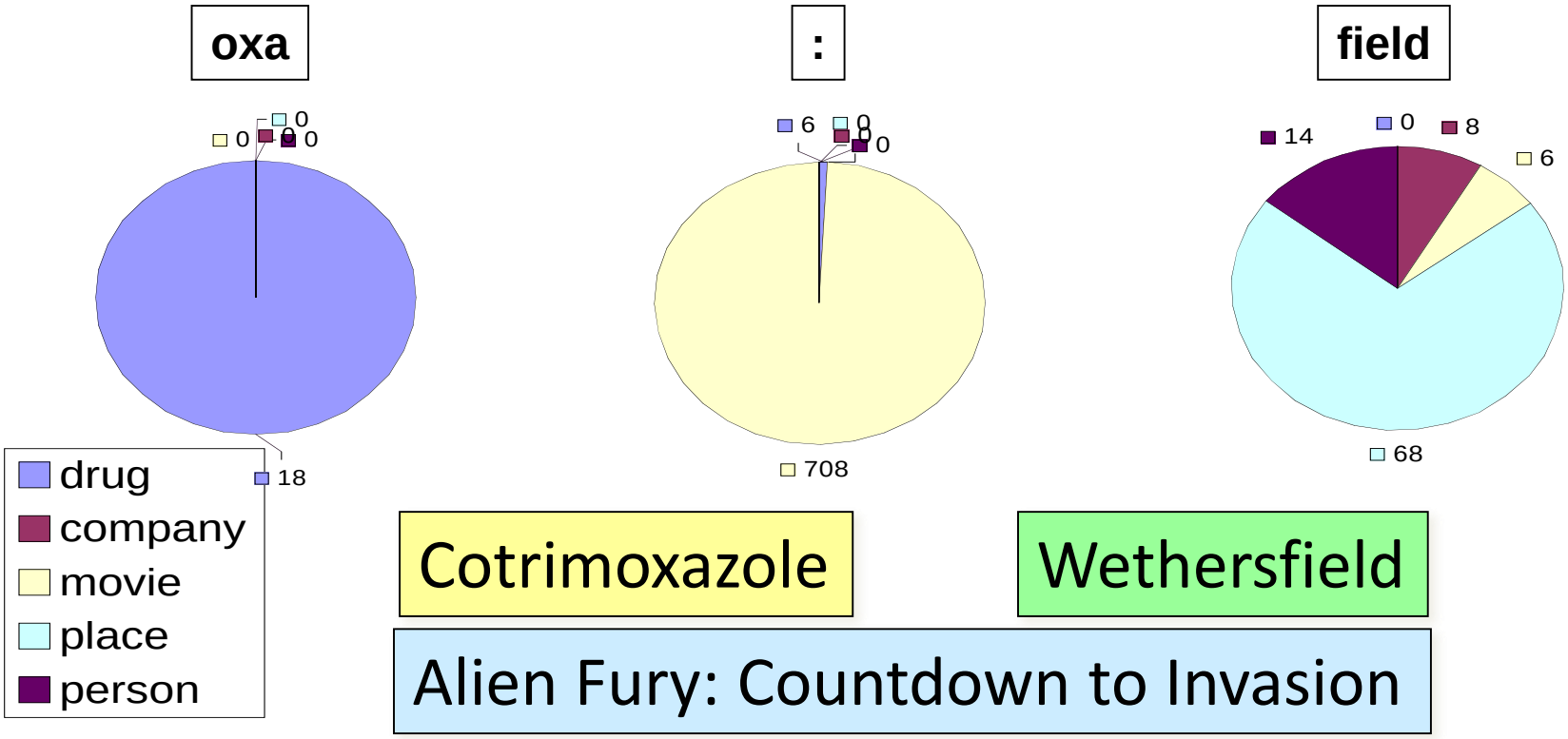
Features for MEMMs

- Goal: to define P(Y|X) using features
- Feature is a function ϕ: H x Y → R
  - often indicators (H x Y → {0,1})
- Each tagging takes input a feature vector

Features for sequence labeling

- Words
  - Current word (essentially like a learned dictionary)
  - Previous/next word (context)
- Other kinds of inferred linguistic classification
  - Part-of-speech tags
- Label context
  - Previous labels

Features: Word substrings



Features: Word shapes

- Word Shapes
  - Map words to simplified representation that encodes attributes such as length, capitalization, numerals, Greek letters, internal punctuation, etc.

Varicella-zoster	Xx-xxx
mRNA	xXXX
CPA1	XXXd

- Common: all prefixes/suffixes of length ≤ 4

Other Features

- N-gram: Unigram, bigram and trigram token sequences in the context window of the current token
- Part-of-Speech: POS tags of the context words
- Gazetteers: person names, organizations, countries and cities, titles, idioms, etc.
- Word clusters: to reduce sparsity, using word clusters such as Brown clusters (Brown et al., 1992)
- Case and Shape: Capitalization and morphology analysis based features
- Chunking: NP and VP Chunking tags
- Global feature: Sentence level and document level features. For example, whether the token is in the first sentence of a document
- Conjunction: Conjunctions of various features

Decoding in MEMMs

MEMM Model

$$P_{MEMM}(y_t \mid \vec{x}; \vec{w}) = \frac{e^{\vec{w}.\vec{\phi}(h_t, y_t)}}{\sum_{y'} e^{\vec{w}.\vec{\phi}(h_t, y')}}$$

- where h_t is the history at position t

- Decoding Problem: most-likely tag sequence

$$\vec{y}^* = \arg\max_{\vec{y}} \quad P(\vec{y} \mid \vec{x}; \vec{w})$$

Training

- Find weights such that

$$LL(\vec{w}) = \sum_{t=1}^{T+1} \log P_{MEMM}(y_t \mid \vec{x}; \vec{w}) - \frac{\lambda}{2} \| \vec{w} \|^2$$

is maximized

Decoding

- Input: $x_1 \ldots x_T$
  - Histories for each position
- Input: features $\phi_1 \ldots \phi_m$
- Input: $w_1 \ldots w_m$
- Output $\vec{y}^* = \arg\max_{\vec{y}} P(\vec{y} \mid \vec{x}; \vec{w})$
- Model

$$P(\vec{y} \mid \vec{x}; \vec{w}) = \prod_{t=1}^{T+1} P_{MEMM}(y_t \mid h_t; \vec{w}) = \prod_{t=1}^{T+1} P_{MEMM}(y_t \mid y_{t-1}, y_{t-2}, \vec{x}, t; \vec{w})$$

$$P_{MEMM}(y_t \mid h_t; \vec{w}) = \frac{e^{\vec{w}.\vec{\phi}(h_t, y_t)}}{\sum_{y'} e^{\vec{w}.\vec{\phi}(h_t, y')}}$$

Dynamic Prog. (Trigram MEMMs)

- First consider how to compute max
- Define $\delta_i(y_{i-1}, y_i) = \max_{y[1:i-2]} P(y_{[1..i]} \mid x_{[1..T]}; \vec{w})$
  - probability of ***most likely*** state sequence ending with tags y_{i-1}, y_i, at position i, given observations $x_1, \ldots, x_T$

$$\delta_i(y_{i-1}, y_i) = \max_{y[1:i-2]} P_{MEMM}(y_i \mid h_i; \vec{w}) P(y_{[1..i-1]} \mid \vec{x}; \vec{w})$$

$$= \max_{y[1:i-2]} P_{MEMM}(y_i \mid y_{i-2}, y_{i-1}, \vec{x}, i; \vec{w}) P(y_{[1..i-1]} \mid \vec{x}; \vec{w})$$

$$= \max_{y_{i-2}} P_{MEMM}(y_i \mid y_{i-2}, y_{i-1}, \vec{x}, i; \vec{w}) \max_{y[1..i-3]} P(y_{[1..i-1]} \mid \vec{x}; \vec{w})$$

$$= \max_{y_{i-2}} P_{MEMM}(y_i \mid y_{i-2}, y_{i-1}, \vec{x}, i; \vec{w}) \delta_{i-1}(y_{i-2}, y_{i-1})$$

Viterbi Algo for Trigram MEMMs

- Initialize: $\delta_0(<s>,<s>) = 1$
- For k=1 to T do
  - For (y',y'') in all possible tagset

$$\delta_i(y', y'') = \max_y P_{MEMM}(y''| y, y', \vec{x}, k; \vec{w})\delta_{i-1}(y, y')$$

$$bp_i(y', y'') = \arg\max_y P_{MEMM}(y''| y, y', \vec{x}, k; \vec{w})\delta_{i-1}(y, y')$$

- Set $y_{T-1}, y_T = \arg\max_{y',y''} P_{MEMM}(</s>| y', y'', \vec{x}, T+1; \vec{w})\delta_T(y', y'')$
- For k=T-2 to 1 do
  - Set $y_k = bp_k(y_{k+1}, y_{k+2})$
- Return y[1..T]

Non-local features & Knowledge for NER

Non-local Features

- Identical tokens should have identical label assignments
  - one tag per discourse!

- Counterexample
  - “Australia” (LOC)
  - “The Bank of Australia” (ORG)

- Approaches (suitable for greedy/beam search decoding)
  - Context Aggregation
  - Two-stage Prediction Aggregation
  - Extended Prediction History

Algorithms

Algorithm	Baseline system	Final System
Greedy	83.29	90.57
Beam size=10	83.38	90.67
Beam size=100	83.38	90.67
Viterbi	83.71	N/A

- Viterbi can't be used with non-local features

Context Aggregation & Two-Stage Prediction

- Augment history of a token by
  - aggregating contexts from all occurences of a word
- May result in excessive number of features
- Two-stage prediction
  - use a baseline NER system for first level predictions
  - use prev predictions as features for final prediction

Not All Mentions Made Equal

- Start of a document more important. Why?
  - Often full name mentioned
  - Match gazetteers better
- Introduce prediction-history feature
  - Aggregate feature counting number of times past tokens (same word) were given a certain tag
  - Left to right decoding

Experiments

	Component	CoNLL03 Test data	CoNLL03 Dev data	MUC7 Dev	MUC7 Test	Web pages
1)	Baseline	83.65	89.25	74.72	71.28	71.41
2)	(1) + Context Aggregation	85.40	89.99	**79.16**	71.53	70.76
3)	(1) + Extended Prediction History	**85.57**	**90.97**	78.56	**74.27**	72.19
4)	(1)+ Two-stage Prediction Aggregation	85.01	89.97	75.48	72.16	**72.72**
5)	All Non-local Features (1-4)	86.53	90.69	81.41	73.61	71.21

Knowledge-based NER

- Is Machine Learning necessary?
  - Just do dictionary lookup
    - only 71.91 F1 on CONLL'03

- Use of gazetteers valuable for ML algorithms
  - use existing gazetteers (loc, census data, etc)
  - mine gazetteers from Wikipedia/Freebase
  - auto-construct gazetteers using Web search
  - matches against each gazetteer a different feature

Obtaining Gazetteers Automatically

- Achieving really high performance for NER requires
  - deep semantic knowledge
  - large costly hand-labeled data

- Many systems also exploited lexical gazetteers
  - but knowledge is relatively static, expensive to construct, and doesn't include any probabilistic information.

Obtaining Gazetteers Automatically

- Data is Power
  - Web is one of the largest text corpora: however, web search is slooooow (if you have a million queries).
- N-gram data: compressed version of the web
  - Already proven to be useful for language modeling

- Patterns over n-grams
  - To autoconstruct gazetteers

Example: Counts on N-grams

died in (a|an) ___ accident

car 13966, automobile 2954, road 1892, auto 1650, traffic 1549, tragic 1480, motorcycle 1399, boating 823, freak 733, drowning 438, vehicle 417, hunting 304, helicopter 289, skiing 281, mining 254, train 250 airplane 236, plane 234, climbing 231, bus 208, motor 198, industrial 187, swimming 180, training 170, motorbike 155, aircraft 152, terrible 137, riding 136, bicycle 132, diving 127, tractor 115, construction 111, farming 107, horrible 105, one-car 104, flying 103, hit-and-run 99, similar 89, racing 89, hiking 89, truck 86, farm 81, bike 78, mine 75, carriage 73, logging 72, unfortunate 71, railroad 71, work-related 70, snowmobile 70, mysterious 68, fishing 67, shooting 66, mountaineering 66, highway 66, single-car 63, cycling 62, air 59, boat 59, horrific 56, sailing 55, fatal 55, workplace 50, skydiving 50, rollover 50, one-vehicle 48, <UNK> 48, work 47, single-vehicle 47, vehicular 45, kayaking 43, surfing 42, automobile 41, car 40, electrical 39, ATV 39, railway 38, Humvee 38, skating 35, hang-gliding 35, canoeing 35, 0000 35, shuttle 34, parachuting 34, jeep 34, ski 33, bulldozer 31, aviation 30, van 30, bizarre 30, wagon 27, two-vehicle 27, street 27, glider 26, " 25, sawmill 25, horse 25, bomb-making 25, bicycling 25, auto 25, alcohol-related 24, snowboarding 24, motoring 24, early-morning 24, trucking 23, elevator 22, horse-riding 22, fire 22, two-car 21, strange 20, mountain-climbing 20, drunk-driving 20, gun 19, rail 18, snowmobiling 17, mill 17, forklift 17, biking 17, river 16, motorcyle 16, lab 16, gliding 16, bonfire 16, apparent 15, aeroplane 15, testing 15, sledding 15, scuba-diving 15, rock-climbing 15, rafting 15, fiery 15, scooter 14, parachute 14, four-wheeler 14, suspicious 13, rodeo 13, mountain 13, laboratory 13, flight 13, domestic 13, buggy 13, horrific 12, violent 12, trolley 12, three-vehicle 12, tank 12, sudden 12, stupid 12, speedboat 12, single 12, jousting 12, ferry 12, airplane 12, unrelated 11, transporter 11, tram 11, scuba 11, common 11, canoe 11, skateboarding 10, ship 10, paragliding 10, paddock 10, moped 10, factory 10

Experiments (CONLL'03)

- Only gazetteer : 71.91
- ML Baseline : 83.65
- Baseline+Gazetteers : 87.22
- Baseline+Gazetteers+Brown : 88.55
- Baseline+Gazetteers+Brown+Non-local : 90.57

	Component	CoNLL03 Test data	CoNLL03 Dev data	MUC7 Dev	MUC7 Test	Web pages
1)	Baseline	83.65	89.25	74.72	71.28	71.41
2)	(1) + External Knowledge	88.55	92.49	84.50	83.23	74.44
3)	(1) + Non-local	86.53	90.69	81.41	73.61	71.21
4)	**All Features**	**90.57**	**93.50**	**89.19**	**86.15**	**74.53**
5)	All Features (train with dev)	90.80	N/A	89.19	86.15	74.33

Unsupervised Name Tagger

Patterns for Gender and Animacy Discovery

Property	Name	*target [#]*	context	*Pronoun*	Example
Gender	Conjunction-Possessive	noun[292,212] |capitalized [162,426]	conjunction	his|her|its|their	*John* and *his*
	Nominative-Predicate	noun [53,587]	am|is|are| was|were|be	he|she|it|they	*he* is John
	Verb-Nominative	noun [116,607]	verb	he|she|it|they	*John* thought *he*
	Verb-Possessive	noun [88,577]| capitalized [52,036]	verb	his|her|its|their	*John* bought *his*
	Verb-Reflexive	noun [18,725]	verb	himself|herself| itself|themselves	*John* explained *himself*
Animacy	Relative-Pronoun	(noun|adjective) & not after (preposition| noun|adjective) [664,673]	comma| empty	who|which| where|when	*John*, *who*

Lexical Property Mapping

Property	Pronoun	Value
Gender	his|he|himself	masculine
	her|she|herself	feminine
	its|it|itself	neutral
	their|they|themselves	plural
Animacy	who	animate
	which|where|when	non-animate

Gender Discovery Examples

- If a mention indicates male and female with high confidence, it's likely to be a person mention

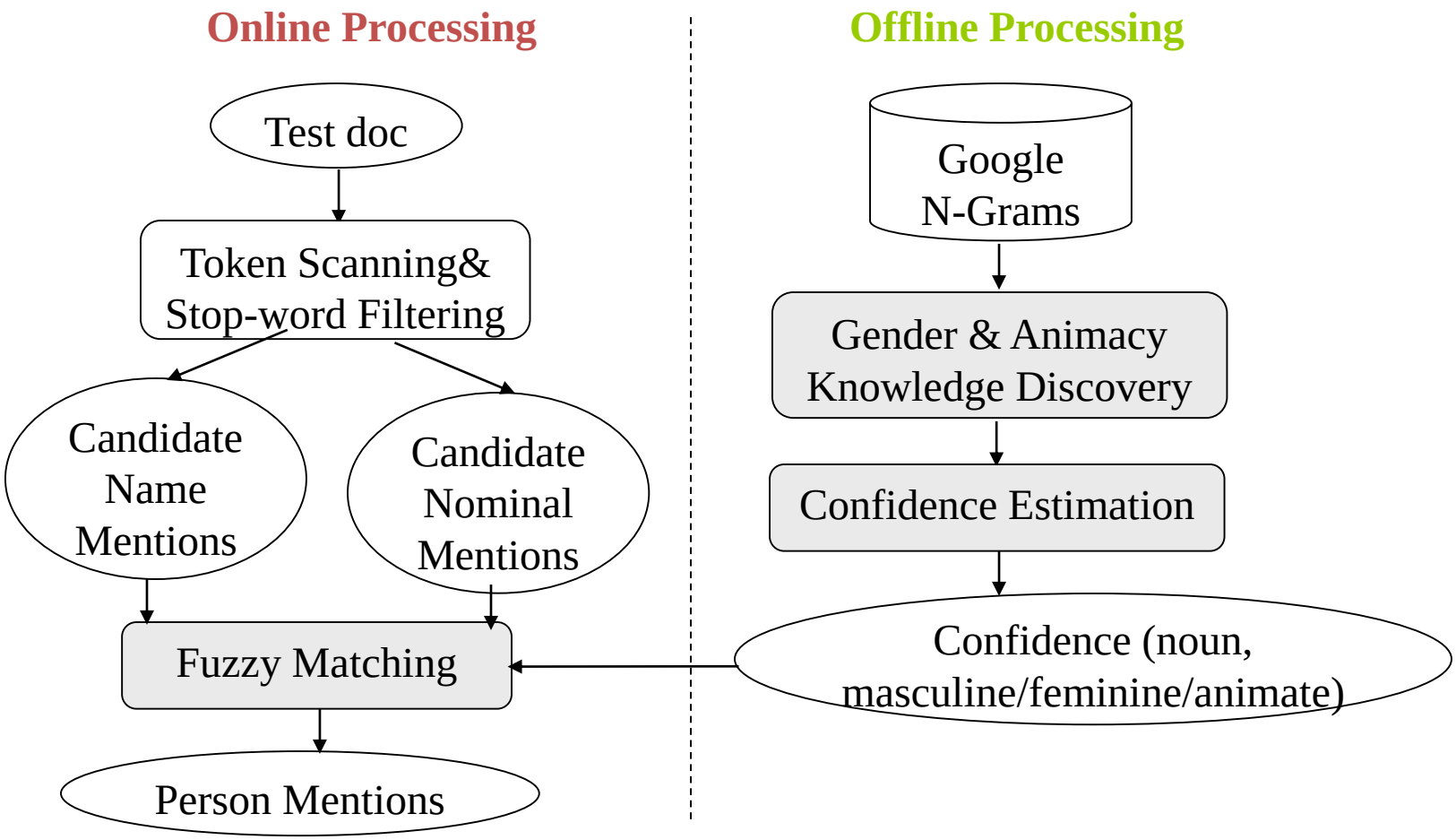
Patterns for candidate mentions	male	female	neutral	plural
John Joseph bought/… his/…	32	0	0	0
Haifa and its/…	21	19	92	15
screenwriter published/… his/…	144	27	0	0
it/… is/… **fish**	22	41	1741	1186

Animacy Discovery Examples

- If a mention indicates animacy with high confidence, it's likely to be a person mention

Patterns for candidate mentions	Animate	Non-Animate		
	who	when	where	which
supremo	24	0	0	0
shepherd	807	24	0	56
prophet	7372	1066	63	1141
imam	910	76	0	57
oligarchs	299	13	0	28
sheikh	338	11	0	0

Overall Procedure



Unsupervised Mention Detection Using Gender and Animacy Statistics

- Candidate mention detection
  - Name: capitalized sequence of <=3 words; filter stop words, nationality words, dates, numbers and title words
  - Nominal: un-capitalized sequence of <=3 words without stop words
- Margin Confidence Estimation

$$\frac{\textit{freq (best property)} - \textit{freq (second best property)}}{\textit{freq (second best property)}}$$

- Confidence (candidate, Male/Female/Animate) > δ
  - **Full matching:** candidate = full string
  - **Composite matching:** candidate = each token in the string
  - **Relaxed matching:** Candidate = any two tokens in the string

Property Matching Examples

Mention candidate	Matching Method	String for matching	Property Frequency			
			masculine	feminine	neutral	plural
John Joseph	Full Matching	John Joseph	**32**	**0**	0	0
Ayub Masih	Composite Matching	Ayub	**87**	**0**	0	0
		Masih	**117**	**0**	0	0
Mahmoud Salim Qawasmi	Relaxed Matching	Mahmoud	**159**	**13**	0	0
		Salim	**188**	**13**	0	0
		Qawasmi	0	0	0	0

Separate Wheat from Chaff: Confidence Estimation

- Rank the properties for each noun according to their frequencies: $f_1 > f_2 > \ldots > f_k$

$$percentage = \frac{f_1}{\sum_{i=1}^{k} f_i}$$

$$margin = \frac{f_1 - f_2}{f_2}$$

$$margin\ \&\ frequency = \frac{f_1}{f_2} \times \log(f_1)$$

Experiments: Data

- Candidate mention detection
  - Name: capitalized sequence of <=3 words; filter stop words, nationality words, dates, numbers and title words
  - Nominal: un-capitalized sequence of <=3 words without stop words
- Margin Confidence Estimation

$$\frac{freq\ (best\ property) - freq\ (second\ best\ property)}{freq\ (second\ best\ property)}$$

- Confidence (candidate, Male/Female/Animate) > δ
  - **Full matching:** candidate = full string
  - **Composite matching:** candidate = each token in the string
  - **Relaxed matching:** Candidate = any two tokens in the string

Impact of Knowledge Sources on Mention Detection for Dev Set

Patterns applied to ngrams for Name Mentions		P(%)	R(%)	F(%)
Conjunction-Possessive	John and his	68.57	64.86	66.67
+Verb-Nominate	John thought he	69.23	72.97	71.05
+Animacy	John, who	85.48	81.96	83.68

Patterns applied to ngrams for Nominal Mentions		P(%)	R(%)	F(%)
Conjunction-Possessive	writer and his	78.57	10.28	18.18
+Predicate	He is a writer	78.57	20.56	32.59
+Verb-Nominate	writer thought he	65.85	25.23	36.49
+Verb-Possessive	writer bought his	55.71	36.45	44.07
+Verb-Reflexive	writer explained himself	64.41	35.51	45.78
+Animacy	writer, who	63.33	71.03	66.96

Overall Experiments

Task	Method	Precision (%)	Recall (%)	F-Measure (%)
Name Mention Detection	Supervised Learning	88.24	81.08	84.51
	Unsupervised Learning Using Knowledge Discovery from Web-scale N-Grams	87.05	82.34	84.63
Nominal Mention Detection	Supervised Learning	85.93	70.56	77.49
	Unsupervised Learning Using Knowledge Discovery from Web-scale N-Grams	71.20	85.18	77.57

Errors:

J P Morgan – recognized as Person